AIAA
Flight Simulation Technologies Conference

A
COLLECTION
OF
TECHNICAL
PAPERS

Scottsdale, AZ
August 1-3, 1994

For permission to copy or republish, contact:
American Institute of Aeronautics and Astronautics
The Aerospace Center
370 L'Enfant Promenade, SW
Washington, DC 20024-2518

A COLLECTION OF TECHNICAL PAPERS

AIAA
Flight Simulation Technologies Conference

August 1-3, 1994/Scottsdale, AZ

ISBN
1-56347-089-6

Printed in the U.S.A.

AIAA Flight Simulation Technologies Conference

General Chair
BRIAN GOLDIEZ
University of Central Florida

Technical Program Chair
THOMAS GALLOWAY
NTSC

The Phoenician
Scottsdale, AZ August 1–3, 1994

Monday Morning / August 1, 1994

Plenary Session — **Precision Orbit Determination: Advances and Applications (8:00 am – 9:00 am)** — **Estrella Theatre**

Welcome: Les Tennan, Member, Arizona Space Commission

Speaker: Byron D. Tapley, The Clare Cockrell Williams Centennial Chair; and Director, Center of Space Research; University of Texas, Austin, TX

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 1-GNC-1 Flight Control Theory Chaired by: D. MELDRUM, University of Washington, Seattle, WA **Ballroom A**							
AIAA-94-3540 **Design of a Digital Flight Control system Using PIM Method** Y. Ochi, T. Ieko, and K. Kanai, *National Defense Academy, Yokosuka, Japan*	**AIAA-94-3541** **Flight Control Design with Input Saturation: MIMO Systems with Stable Plants** R. Hess, *Univ. of California, Davis, CA*	**AIAA-94-3542** **Robust Multirate Eigenstructure Assignment with Flight Control Application** J. Piou and K. Sobel, *City College of New York, New York, NY*	**AIAA-94-3545** **Development of an Air-to-Air Refueling Automatic Flight Control System Using Quantitative Feedback Theory** D. Trosen, M. Pachter, and C. Houpis, *Air Force Inst. of Technology, Wright-Patterson AFB, OH*	**AIAA-94-3544** **Control System Analysis in Nonlinear Flight Regimes** B. Allan, A. Packard, and C. Atwood, *Univ. of California, Berkeley, CA*	**AIAA-94-3543** **Discrete Sliding Mode Control of Wing Rock** J. Fernand and D. Downing, *USAF Academy, Colorado Springs, CO*		
Session 2-GNC-2 Navigation, Tracking, and Estimation: Theory and Analysis Chaired by: M. PSIAKI, Cornell University, Ithaca, NY **Ballroom B**							
AIAA-94-3546 **Modified Spherical Coordinates for Radar** P. Robinson and M. Yin, *Litton Data Systems, Agoura, CA*	**AIAA-94-3547** **Kalman Filter Inertial Navigation System Error Model based on Filter Stability Considerations** R. Rogers, *Rogers Engineering and Associates, Gainesville, FL*	**AIAA-94-3548** **General Methods of Estimate Fusion Applied to Spacecraft Rendezvous** J. Carpenter and R. Bishop, *NASA Johnson, Houston, TX*	**AIAA-94-3549** **Distributed Multisensor Fusion** L. Pao, *Northwestern Univ., Evanston, IL*	**AIAA-94-3550** **A Real-Time Model Error Filter and State Estimator** J. Crassidis and L. Markley, *NASA Goddard, Greenbelt, MD;* D. Mook, *SUNY at Buffalo, Buffalo, NY*	**AIAA-94-3551** **A Process Noise Covariance Estimator** P. Mason and D. Mook, *SUNY at Buffalo, Buffalo, NY*		
Session 3-GNC-3 Aircraft Control and Guidance Applications Chaired by: W. ARNOLD III, Sverdrup Technology, Inc., Eglin AFB, FL **Ballroom F**							
AIAA-94-3552 **Closed-Form Solutions to the Constrained Control Allocation Problem** K. Bordi and W. Durham, *VPI, Blacksburg, VA*	**AIAA-94-3553** **Longitudinal Axis Display Requirements for High Speed Cruise** D. Honaker and M. Anderson, *VPI, Blacksburg, VA*	**AIAA-94-3554** **Optimal and Suboptimal Minimum Time-to-Climb Trajectories** H. Sewald, *Analytical Mechanics Associates, Inc., Hampton, VA*	**AIAA-94-3555** **Long Flight-Time Range-Optimal Atmospheric Flight Vehicle Trajectories** H. Sewald, *Analytical Mechanics Associates, Inc., Hampton, VA*	**AIAA-94-3556** **Aircraft Time-Optimal Heading-Reversal Maneuvers** S. Bocvarov, E. Cliff, and F. Lutze, *VPI, Blacksburg, VA*	**AIAA-94-3557** **Formation Flight Control Automation** V. Reyna, M. Pacher, and J. D'Azzo, *Air Force Inst. of Technology, Wright-Patterson AFB, OH*		
Session 4-GNC-4 Spacecraft Attitude Determination a and Control I Chaired by: G. SEVASTON, Jet Propulsion Laboratory, Pasadena, CA **Ballroom G**							
AIAA-94-3558 **Magnetic Precession of Bias Momentum Satellites** H. Hablani, *Rockwell International, Seal Beach, CA*	**AIAA-94-3559** **Exact Singularity Avoidance Control of the Pyramid Type CMG System** H. Kurokawa, *Mechanical Engineering Lab, Ibaraki, Japan*	**AIAA-94-3560** **A High Performance Robust Attitude Controller for Flexible Spacecraft** B. Appleby, *C. S. Draper Lab, Cambridge, MA*	**AIAA-94-3561** **Attitude Determination Studies for EOS-AM1** P. Kudva, *McDonnell Douglas Aerospace, Seabrook, MD*	**AIAA-94-3562** **An Object Oriented Attitude Control Design for Saturn/Titan Exploration** E. Wong, *JPL, Pasadena, CA*	**AIAA-94-3563** **Feedback Control Logic for Spacecraft Eigenaxis Rotations Under Slew Rate and Control Constraints** B. Wie and J. Lu, *Arizona State Univ., Tempe, AZ*		

Monday Morning / August 1, 1994

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 5-GNC-5 Missile Guidance Chaired by: J. BIBEL, Naval Surface Warfare Center, Dahlgren, VA							**Room 7**
AIAA-94-3564 **Explicit Guidance Using Collocation Methods for Reentry Vehicles** E. Ferreira and P. Delpy, *Aerospatiale, Les Mureaux, France*	**AIAA-94-3565** **New Guidance Law for a Missile with Varying Velocity** Y. Baba, T. Takehira, and H. Takano, *National Defense Academy, Yokosuka, Japan*	**AIAA-94-3566** **Two-Variable-Structure Homing Guidance Schemes With and Without Target Maneuver Estimation** K. Babu, I. Sarma, and K. Swamy, *Indian Inst. of Science, Bangalore, India*	**AIAA-94-3567** **Linear-Quadratic Pursuit-Evasion Games with Terminal Velocity Constraints** J. Ben-Asher, *Tel Aviv Univ., Tel Aviv, Israel*	**AIAA-94-3568** **Ascent Guidance Comparisons** J. Hanson, M. Shrader, and C. Cruzen, *NASA Marshall, Huntsville, AL*			
Session 6-GNC-6 Flexible Structure Control I Chaired by: H. Fujii, Tokyo Metropolitan Institute of Technology, Tokyo, JAPAN							**Room 8**
AIAA-94-3569 **Robust Time-Optimal Control of Flexible Structures: A Frequency Domain Approach** T. Singh and S. Vadali, *SUNY at Buffalo, Buffalo, NY*	**AIAA-94-3570** **Robustness Verification of Flexible Structures via Interval Control Systems** T. Link, J-S. Lew, and L. Keel, *Tennessee State Univ., Nashville, TN*	**AIAA-94-3571** **A Comparison of Robust Control Techniques for Uncertain Control Systems** S. Grocott, J. How, and D. Miller, *MIT, Cambridge, MA*	**AIAA-94-3572** **Beam-Waveguide Antenna Servo Design Issues for Tracking Low-Earth-Orbiting Satellites** W. Gawronski and J. Mellstrom, *JPL, Pasadena, CA*	**AIAA-94-3573** **An Optimal Bounded-State Controller for Flexible Structures** C-H. Chuang and D-N. Wu, *Georgia Inst. of Technology, Atlanta, GA*			
Session 7-GNC-7 Computational Control and Dynamics Chaired by: J. SUNKEL, NASA Johnson Space Center, Houston, TX							**Room 9**
AIAA-94-3574 **Dynamic Modal Truncation** S. Djerassi, *Nahariya, Israel*	**AIAA-94-3575** **Motion Calibration/ Identification of a Flexible Manipulator** A. Barnes, *Univ. of Kentucky, Lexington, KY*	**AIAA-94-3576** **Order-n Formulation of Equations of Motion with Efficient Choices of Motion Variables** A. Banerjee, *Lockheed Palo Alto Research Lab, Palo Alto, CA*	**AIAA-94-3577** **Comparison of Simulation with Test of Deployment of a Wrapped-Rib Antenna** M. Lemak, *Lockheed Missiles & Space Co., Sunnyvale, CA*	**AIAA-94-3578** **Symbolic Equation Processing and Automated Code Generation for Computational Control** R. Singh, *Dynacs Engineering Co., Palm Harbor, FL*	**AIAA-94-3579** **Parallel Computing of Massively Actuated Large Structures: Performance Assessment and Challenges** K. Park, *Center for Aerospace Structures, Boulder, CO*		
Session 8-AFM-1 Missiles, Projectiles, and Store Separation Chaired by: T. JORDAN, Sandia National Laboratories, Albuquerque, NM							**Room 1**
AIAA-94-3450 **Missile Sizing for Boost-Phase Intercept** D. Hull, *Univ. of Texas at Austin, Austin, TX;* D. Salguero, *Sandia National Labs, Albuquerque, NM*	**AIAA-94-3451** **Multiple Flexible Body Missile Launcher Simulation** J. Cochran Jr., Y-M. Cheng, and S. Bigelow, *Auburn Univ., Auburn, AL;* D. Sandidge, *U.S. Army Missile Command, Redstone Arsenal, AL*	**AIAA-94-3452** **An Investigation of Spin-Inducing Afterbody Configurations for Hypervelocity Projectiles** D. Barnette, *Univ. of Texas at Austin, Austin, TX*	**AIAA-94-3453** **Hypervelocity Projectile Flight Characteristics Including In-Flight Deployment and Shape Change** H. Legner and E. Lo, *Physical Sciences, Inc., Andover, MA;* W. Reinecke, *Univ. of Texas, Austin, TX*	**AIAA-94-3454** **Separation Analysis of the Pegasus XLK From an L-1011 Aircraft** W. Sickles, M. Rist, and C. Morgret, *Calspan Corp., Arnold AFB, TN;* K. Parthasarathy, *Johns Hopkins Univ., Laurel, MD*	**AIAA-94-3455** **Computer Simulation of Store Separation by a Transonic Small-Disturbance Theory** C. Lan, J. Luo, and C. Hsing, *Univ. of Kansas, Lawrence, KS;* S. Chin and Y. Chen, *Aero Industry Development Center, Taiwan, ROC*		

Monday Morning / August 1, 1994

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 9-AFM-2 Stability and Control I Chaired by: J. FERNAND,USAF Academy, Colorado Springs, CO							**Room 2**
AIAA-94-3456 **Optimizing the Use of Multiple Control Effectors Using Bifurcation Analysis** M. Lowenberg, *Univ. of Bristol, Bristol, UK*	**AIAA-94-3457** **Measurement of Aircraft Stability Parameters in the Wind Tunnel using a Wind Driven Manipulator** J. Magill, L. Darden, N. Komerath, and J. Dorsey, *Georgia Inst. Technology, Atlanta, GA*	**AIAA-94-3458** **The Nonlinear Indicial Response: Vis-a-Vis Roll-Rate Induced Camber Effects** J. Jenkins, *Wright Lab, Wright-Patterson AFB, OH*	**AIAA-94-3513** **VISTA/F-16 Multi-Axis Thrust Vectoring (MATV) Control Law Design and Evaluation** W. Zwerneman and B. Eller, *Lockheed Fort Worth Co., Fort Worth, TX*				
Session 10-AFM-3 Hypersonics and Entry Technology Chaired by: W. RUTLEDGE, Sandia National Laboratories, Albuquerque, NM							**Room 3**
AIAA-94-3461 **Hypersonic Research Program at NASA Ames Research Center (1hr invited)** M. Green, *NASA Ames, Moffett Field, CA*		**AIAA-94-3462** **General Purpose Heat Source Module Hypersonic Reentry Attitude Stability** D. Platus, *The Aerospace Corporation, Los Angeles, CA*	**AIAA-94-3463** **Flight Dynamic and Aerothermodynamic Conceptual Design of a Bent-Nose Biconic RV Bus for Atmospheric Entry From Low Earth Orbit** J. White, *Hughes Missile Systems Co., Pomona, CA*	**AIAA-94-3465** **Sizing of a Controlled Retrorockets System for the Soft Landing of a Reentry Vehicle** T. Abensur, *Aerospatiale, Les Mureaux, France*			
Session 11-AFM-4 Aircraft Performance Chaired by: S. MORRIS, USAF Academy, Colorado Springs, CO							**Room 5**
AIAA-94-3466 **A Method for Predicting Combined Dorsal Normal Force and Center-of-Pressure Location at Supersonic Mach Numbers** J. White, *Hughes Missile Systems Co., Pomona, CA*	**AIAA-94-3467** **An Object-Oriented Approach to Aircraft Performance Analysis** M. Yokell, *Lockheed Fort Worth Co., Fort Worth, TX;* T. Baker, *Computer Sciences Corp., Fort Worth, TX*	**AIAA-94-3468** **Weapon Drag Coefficient Determination Using Genetic Algorithm** M. Anderson, *Sverdrup Technology, Eglin AFB, FL;* R. McCurdy, *USAF, Eglin AFB, FL*	**AIAA-94-3469** **Optimum Supersonic Climb** N. Vinh, *Univ. of Michigan, Ann Arbor, MI;* Y. Tzeng, *Chung Shan Inst. of Science and Technology, Lungtan, Taiwan, ROC*				
Session 12-FST-1 Simulation Modeling Chaired by: J. MCCRILLIS, Naval Air Warfare Center, Patuxent River, MD							**Ballroom E**
AIAA-94-3400 **Improvements to the Naval Air Warfare Center Aircraft Division's F/A-18 Subsonic Aerodynamic Model** T. Fitzgerald, *Naval Air Warfare Center, Patuxent River, MD;* J. Ralston, *Bihrle Applied Research, Inc., Hampton, VA;* B. Hildreth, *Systems Control Technology, Inc., Lexington Park, MD*	**AIAA-94-3401** **Comparison of Frequency Response and Perturbation Methods to Extract Linear Models from a Nonlinear Simulation** K. Balderson and J. Weathers, *Naval Air Warfare Center, Patuxent River, MD*	**AIAA-94-3402** **Matching a Flight Training Device with a Real Aircraft Using a Genetic Algorithm** R. Braunstingl, *Technical Univ. of Graz, Graz, Austria*	**AIAA-94-3403** **Optimal Simulator Aerodynamic Model Definition Using a Genetic Algorithm** D. Hensley, *Boeing Commercial Airplanes, Kent, WA*	**AIAA-94-3404** **Aspects of C-160 Simulator Model Determination and Validation On and Close to the Ground** D. Fischenberg, W. Monnich, B. Krag, and R. Jategaonkar, *DLR, Braunschweig, Germany*			

Monday Morning / August 1, 1994

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 13-FST-2 Human Factors Chaired by: F. CARDULLO, Binghamton University, Binghamton, NY							**Room 6**
AIAA-94-3405 **Open and Closed Loop Control with a Perspective Tunnel-in-the-Sky Display** E. Theunissen, *Delft Univ. of Technology, Delft, The Netherlands*	**AIAA-94-3406** **Choice and Judgment: The Future of Flight Training** S. Hayne, *Univ. of Calgary, Calgary, Alberta, Canada;* C. Smith, *Univ. of Montana, Missoula, MT*	**AIAA-94-3407** **General Human Machine Interface Requirements for Avionics Systems Design** S. Gikas and P. Markopoulos, *Univ. of London, London, UK*					
Session 14-ASD-1 Orbit Analysis I Chaired by: S. ALFANO, Phillips Laboratory, Kirtland AFB, NM							**Room 10**
AIAA-94-3704 **The Computation of Satellite Constellation Range Characteristics** K. Gordon, *The Aerospace Corp., Los Angeles, CA*	**AIAA-94-3705** **Data Parallel Implementation of Brouwer-Lyddane Theory: FORTRAN 90, Connection Machine FORTRAN, and Matlab** D. Carter, *C. S. Draper Lab, Cambridge, MA*	**AIAA-94-3706** **Performance of Analytic Orbit Propagators on a Hypercube and a Workstation Cluster** B. Nata, D. Danielson, S. Ostrom, and S. Brewer, *Naval Postgraduate School, Monterey, CA*	**AIAA-94-3707** **General Human Machine Interface Requirements for ATM and Avionics System Design** J. Junkins and T. Prasanna, *Texas A&M Univ., College Station, TX*	**AIAA-94-3708** **Dynamical Transformations of a Conservative System** M. Hough, *Textron Defense Systems, Wilmington, MA*	**AIAA-94-3709** **A Computer Demonstration of Onboard Orbit Control Using a GPS Receiver** C. Chao, H. Bemstein, M. Menn, and R. Gist, *The Aerospace Corp., Los Angeles, CA*		
Session 15-ASD-2 Attitude Dynamics Chaired by: J. COCHRAN							**Room 11**
AIAA-94-3711 **The Pitch Dynamics of a Satellite in the Elliptic Problem of Three Bodies** J. Ashenberg, *Chelmsford, MA*	**AIAA-94-3712** **Efficient Estimation of Attitude Sensor Coalignments** M. Shuster, *Univ. of Florida, Gainesville, FL*	**AIAA-94-3713** **Analytic Solution for the Velocity of a Rigid Body During Spinning-Up Maneuvers** R. Beck and J. Longuski, *Purdue Univ., West Lafayette, IN*	**AIAA-94-3714** **Resonance Capture in Dynamically Unbalanced Dual-Spin Spacecraft** C. Hall and R. Tsui, *USAF, Wright-Patterson AFB, OH*	**AIAA-94-3715** **The Motion of a Rigid Body with an Attached Spring-Mass-Damper** C. Hall and A. Chinnery, *USAF, Wright-Patterson AFB, OH*			
Session 16-ASD-3 Mission Design: Earth Orbit and Lunar Chaired by: R. BARNISIN, Orbital Sciences Corporation, Dulles, VA							**Room 12**
AIAA-94-3716 **Lunar Scout Launch Window** L. Wagner Jr., *NASA Johnson, Houston, TX*	**AIAA-94-3717** **Orbital Mission Analysis for a Lunar Mapping Satellite** S-Y. Park and J. Junkins, *Texas A&M Univ., College Station, TX*	**AIAA-94-3718** **A Design Concept for Multiple Lunar Swingby Trajectories** K. Howell and R. Wilson, *Purdue Univ., West Lafayette, IN*	**AIAA-94-3719** **Spacecraft Shadow Impingement onto its Solar Arrays** R. Gist, *The Aerospace Corp., Los Angeles, CA*	**AIAA-94-3720** **Polar Elliptic Orbits for Global Coverage Constellations** F. Graziani and G. Palmerini, *Univ. of Rome, La Sapienza, Italy*	**AIAA-94-3721** **Exact Design of Partial Coverage Satellite Constellations Over Oblate Earth** D-M. Ma and W-C. Hsu, *Tamkang Univ., Tamsui, Taiwan, ROC*		

Monday Afternoon / August 1, 1994

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
Session 17-GNC-8 Control Theory I Chaired by: J. DUPONT, Aerospatiale, Les Mureaux, France							**Ballroom A**
AIAA-94-3580 **Stabilization of Numerical Solutions of Boundary Value Problems Exploiting Invariants** E. Eich, R. Mehlhorn, and G. Sachs, *Technische Univ. Munchen, Munchen, Germany*	**AIAA-94-3581** **Design of Optimal Residuals for Detecting Sensor Faults Using Multi-Objective Optimization and Generic Algorithms** J. Chen, R. Patton, and G. Liu, *Univ. of York, York, UK*	**AIAA-94-3582** **Eigenstructure Perturbation in Disjointed Domains for Linear Systems with Structured Uncertainty** R. Yedavalli and C. Ashokkumar, *Ohio State Univ., Columbus, OH*	**AIAA-94-3583** **A Finite Difference Based Scheme for Automatic Costate Calculation** H. Seywald, *Analytical Mechanics Associates, Inc., Hampton, VA*	**AIAA-94-3584** **Controllability and Observability of Algebraic Equations of Motion by Hamilton's Law of Varying Action** E. Adiguzel and H. Oz, *Ohio State Univ., Columbus, OH*	**AIAA-94-3586** **Fuzzy and Adaptive Control of an Aircraft** J. Sasiadek and A. Mazzawi, *Carleton Univ., Ottawa, Ontario, Canada*		
Session 18-GNC-9 Aircraft Control Applications Chaired by: R. COLGREEN, Lockheed Advanced Development Company, Palmdale, CA							**Ballroom B**
AIAA-94-3587 **An Intelligent Control Paradigm for Flight Control Applications** F. Swern, C. Chassapis, and A. van der Veen, *Stevens Inst. of Technology, Hoboken, NJ*	**AIAA-94-3588** **Non-Interactive Control by Eigenstructure Assignment and Feedforward** T. Livet, F. Kubica, J-F. Magni, and L. Antonel, *Aerospatiale, Toulouse, France*	**AIAA-94-3589** **On Control Power Optimization Using Linear Matrix Inequalities** R. Niewoehner and I. Kaminer, *Naval Postgraduate School, Monterey, CA*	**AIAA-94-3590** **Stability Augmentation Design of a Large Subsonic Transport** G. Ward and U-L. Ly, *Boeing Commercial Airplane Group, Seattle, WA*	**AIAA-94-3591** **Application of Theory of Stability Under Constrain to the Landing Approach Task** D. Moorhouse and T. Kelly, *USAF, Wright-Patterson AFB, OH*	**AIAA-94-3592** **Nonlinear Inverse Dynamics Control of Aircraft Using Spoilers** X. Sun, K. Woodgate, and Allwright, *Imperial College, London, UK*		
Session 19-GNC-10 Flexible Structure Control II Chaired by: W. GAWRONSKI, Jet Propulsion Laboratory, Pasadena, CA							**Ballroom F**
AIAA-94-3593 **A Distributed Parameter System Model for Active Control of a Large, Flexible Phased Array Transmitter** J. Stuart, M. Balas, and S. Nozette, *Univ. of Colorado, Boulder, CO*	**AIAA-94-3594** **Joint Decoupling Methods for Decentralized Control of Flexible Structures** T-J. Su, *Keelung, Taiwan, ROC*	**AIAA-94-3595** **Experimental Research on Control of a Flexible Frame Structure from a Wave Point of View** K. Matsuda and H. Fujii, *Tokyo Metropolitan Inst. of Technology, Tokyo, Japan*	**AIAA-94-3596** **Active Distributed Vibration Control of General Anisotropic Piezo-Laminated Plates** S. Miller, H. Abramovich, and Y. Oshman, *Technion, Haifa, Israel*	**AIAA-94-3597** **Real-Time RMS Active Damping Augmentation: Heavy and Very Light Payload Evaluation** M. Demeo, *ViGYAN, Hampton, VA;* M. Gilbert, *NASA Langley, Hampton, VA;* J. Lepanto, and K. Flueckiger, *Draper Lab, Cambridge, MA;* E. Bains, NASA Johnson, Houston, TX; M. Jensen, *Lockheed Eng. and Sci., Co.,Houston, TX*			
Session 20-GNC-11 Neural Networks Applications Chaired by: G. VUKOVICH, Canadian Space Agency, St. Hubert, Quebec, Canada							**Ballroom G**
AIAA-94-3598 **Sensor Failure Detection, Identification and Accommodation Using On-Line Learning Neural Architectures** M. Napolitano, C. Neppach, V. Casdorph,and S. Naylor, *West Virginia Univ., Morgantown, WV;* M. Innocenti and F. Bini, *Univ. of Pisa, Pisa, Italy*	**AIAA-94-3599** **Neural Network Approach to Aerodynamic Coefficient Estimation and Aircraft Failure Isolation Design** C-Y. Chiang and H. Youssef, *Univ. of Southern California, Los Angeles, CA*	**AIAA-94-3600** **Combining Traditional State Space Control Methods with Neural Networks** B. Smith and J. Adams, *Phillips Lab, Kirtland AFB, NM*	**AIAA-94-3601** **Neural Networks for Performance Improvement in Robot Control During General Task Evaluation** P. Chen, J. Mills, and K. Smith, *Univ. of Toronto, Toronto, Ontario, Canada*	**AIAA-94-3602** **Fuel-Optimal Periodic Control for a Hypersonic Vehicle Using Neural On-Line Training** C. Chuang and T. Lee, *Georgia Inst. of Technology, Atlanta, GA*			

Monday Afternoon / August 1, 1994

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
Session 21-GNC-12 Missile Control and Estimation Chaired by: A. RODRIGUEZ, Arizona State University, Tempe, AZ, and P. RAO, Martin Marietta Corporation, Denver, CO							**Room 7**
AIAA-94-3603 **An Example Application of Analysis for Multivariable Systems on the AGM-130A Missile** R. Stoughton, J. Kelley, and A. Bindemann, *Sverdrup Technology, Eglin AFB, FL*	**AIAA-94-3604** **Missile Flight Control with Dynamic Inversion and Structured Singular Value Synthesis** C. D'Souza and M. McFarland, *Dept. of the Air Force, Eglin AFB, FL*	**AIAA-94-3605** **Terminal Homing Performance of Semi-Active Missiles Against Multi-Target Raids** C. Phillips, *Naval Surface Warfare Center, Dahlgren, VA*	**AIAA-94-3606** **Robust Electro-Optical Seeker for Exo, Endo and Terminal Guidance Applications** S. Smith, *Naval Air Warfare Center, China Lake, CA*	**AIAA-94-3607** **Guidance Law for Homing Missile with Saturation** A. Rodriguez, *Arizona State Univ., Tempe, AZ;* S. Balakrishnan, *Univ. of Missouri-Rolla, Rolla, MO*	**AIAA-94-3608** **Robustness of the Differential Corrections Algorithm in an Air Data Estimation System** M. Anderson, W. Lawrence, and J. Lopez, *Sverdrup Technology, Eglin AFB, FL*		
Session 22-GNC-13 Integrated Flight/Propulsion Control Design with STOVL Aircraft Applications Chaired by: S. BALAKRISHNAN, University of Missouri-Rolla, Rolla, MO							**Room 8**
AIAA-94-3609 **Multivariable Control Allocation and Control Law Conditioning When Control Effectors Limit** J. Virnig and D. Bodden, *Lockheed Fort Worth Co., Fort Worth, TX*	**AIAA-94-3610** **Propulsion Control Aspects of Integrated Flight and Propulsion Control Design** S. Adibhatla, *GE Aircraft Engines, Cincinnati, OH*	**AIAA-94-3611** **Application of anIntegrated Methodology for Propulsion and Airframe Control Design to a STOVL Aircraft** S. Garg, *NASA Lewis, Cleveland, OH;* D. Mattern, *NYMA, Inc., Brook Park, OH*	**AIAA-94-3612** **Piloted Evaluation of an Integrated Methodology for Propulsion and Airframe Control Design** M. Bright, D. Simon, and S. Garg, *NASA Lewis, Cleveland, OH;* D. Mattern, *NYMA, Inc., Brook Park, OH*	**AIAA-94-3613** **Integrated Flight/ Propulsion Control Specifications: Accounting for 2-Way Coupling** K. Neighbors and S. Rock, *Stanford Univ., Stanford, CA*	**AIAA-94-3614** **Performance Limitations of Decentralized Control Laws for Integrated Flight and Propulsion Control** J. Schierman, *Arizona State Univ., Tempe, AZ;* D. Schmidt, *Univ. of Maryland, College Park, MD*		
Session 23-GNC-14 Spacecraft Attitude Determination and Control II Chaired by: F. BAUER, NASA Goddard Space Flight Center, Greenbelt, MD							**Room 9**
AIAA-94-3615 **Suboptimal Command Generation for Control Moment Gyroscopes and Feedback Control of Spacecraft** S. Vadali, *Texas A&M Univ., College Station, TX*	**AIAA-94-3616** **Regional Pole Placement for Momentum Management Controller for the Space Station** D. Naidu, *Idaho State Univ., Pocatello, ID*	**AIAA-94-3617** **Galileo Spacecraft Scan Platform Celestial Pointing Cone Control Gain Redesign** C-H. Ih, *JPL, Pasadena, CA*	**AIAA-94-3618** **A Zero-Gyro, Zero-Wheel Controller for Hubble Space Telescope** G. Sandhoo, *George Washington Univ., Washington, DC*	**AIAA-94-3619** **Star Catalog Development for CT-601 Star Trackers, with Application to EOS-AM1** P. Kudva, *McDonnell Douglas Aerospace, Seabrook, MD*	**AIAA-94-3620** **On the Recursive Formula of Rhumb Line Maneuvers with its Applications** J. Kawaguchi, Y. Morita, H. Yamakawa, and T. Hashimoto, *Inst. of Space and Astronautical Science, Kanagawa, Japan*		
Session 24-AFM-5 Aircraft Dynamics I Chaired by: E. BARLAND, Lockheed Aeronautical Systems Co., Marietta, GA							**Room 1**
AIAA-94-3470 **A Multi-Point Approach for Aerodynamic Modeling in Complex Flow-Fields** P. Jaramillo, Y. Cho, and M. Nagati, *Wichita State Univ., Wichita, KS*	**AIAA-94-3471** **Reformulation of the Acceleration Rate Vector from Frenet's Formulas Using Body Axis Velocities and Accelerations** J. Booz, *Naval Air Warfare Center, Warminster, PA*	**AIAA-94-3472** **Flight Envelopes for Hypersonic Aero-Rendezvous at Constant Altitude** J-S. Chern, *Chung Shan Inst. of Science and Technology, Lungtan, Taiwan, ROC;* Z-C. Hong, Y-H. Chen, and J-M. Liu, *National Central Univ., Chungli, Taiwan, ROC*	**AIAA-94-3473** **Identification of Speed Brake, Ramp Door, Air-Drop, and Landing Gear Effects from "Transall" Flight Data** R. Jategaonkar, W. Monnich, D. Fischenberg, and B. Krag, *DLR, Braunschweig, Germany*	**AIAA-94-3474** **Gust Effect on Fighter Aircraft During Maneuvering Flight** M. Ghazi, *King Abdulaziz Univ., Jeddah, Saudi Arabia*			

Monday Afternoon / August 1,, 1994

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
Session 25-AFM-6 High Angle of Attack Aerodynamics Chaired by: D. SPRING, Auburn University, Auburn, AL							**Room 2**
AIAA-94-3475 **Numerical Analysis of Tangential Slot Blowing on a Generic Chined Forebody** R. Agosta-Greenman and L. Schiff, *NASA Ames, Moffett Field, CA;* K. Gee, *MCAT Inst., Moffett Field, CA;* R. Cummings, *California Polytechnic State Univ., San Luis Obispo, CA*	**AIAA-94-3476** **Wind Tunnel Tests of Full-Scale F/A-18 Twin Tail Buffet: A Summary of Pressure and Response Measurements** C. Pettit, D. Brown, and E. Pendleton, *Wright Lab, Wright-Patterson AFB, OH*	**AIAA-94-3477** **Experimental Investigation of Tangential Slot Blowing on a Generic Chined Forebody** R. Cummings and J. Duino, *California Polytechnic State Univ., San Luis Obispo, CA;* L. Schiff, *NASA Ames, Moffett Field, CA*	**AIAA-94-3478** **Lift Enhancement of a Wing/Strake Configuration Using Pneumatic Blowing** R. Howard, J. Wilson, and C. Zgraggen, *Naval Postgraduate School, Monterey, CA*	**AIAA-94-3479** **Critical States and Flow Topology on a 65 Degree Delta Wing** C. Jobe, A. Hsia, and J. Jenkins, *Wright Lab, Wright-Patterson AFB, OH*			
Session 26-AFM-7 Work-in-Progress Chaired by: G. CHRUSCIEL, Lockheed Missiles and Space, Sunnyvale, CA							**Room 3**
Upgrade to NASA Ames 16-inch Shock Tunnel Capabilities G. Deiwert, *NASA Ames, Moffett Field, CA*	**The Role fo Aeroballistic Ranges for Improved Fire Control and Flight Control** R. Adelgren and C. Howerton, *Wright Lab, Eglin AFB, FL*	**Math Modeling of Aero Data for Aircraft Dynamic Motion** J. Kalviste, *Northrop Corp., Pico Rivera, CA*	**Inlet Unstart Detection in Supersonic Flows** W. Hawkins, *Calspan Corp., Arnold AFB, TN*	**Full Aero-Thermo-Structural Simulation Capabilities for Mach 7** D. Marren and J. Lafferty, *Naval Surface Warfare Center, Silver Spring, MD*	**Recent Developments in LENS Facility** M. Holden, *CUBRC, Buffalo, NY*	**Concept for Virtual Flight Testing** C. Ratliff, *Calspan Corp., Arnold AFB, TN*	**THAAD Kill Vehicle Ground Test Results** V. Eachus, *Lockheed Missiles and Space Co., Sunnyvale, CA*
Session 27-AFM-8 Estimation of Aircraft Stability and Control Parameters Chaired by: K. BILMORIA, Arizona State University, Tempe, AZ							**Room 5**
AIAA-94-3480 **Unsteady Aerodynamic Modeling of a Canard Aircraft for Parameter Estimation** S. Raisinghani and H. Kumar, *Indian inst. of Technology Kanpur, Kanpur, India*	**AIAA-94-3481** **Parameter Estimation for a Cessna U-206 Aircraft Using the Maximum Likelihood Method** M. Napolitano, A. Paris, and B. Seanor, *West Virginia Univ., Morgantown, WV;* A. Bowers, *NASA Dryden, Edwards, CA*	**AIAA-94-3482** **Correlation of the Damping in Pitch Stability Derivatives for Body-Tail Configurations** A. Sigal, *Technion, Haifa, Israel*					
Session 28-FST-3 Computing Hardware and Software Chaired by: D. BROOKS, Boeing Commercial Airplane Company, Seattle, WA							**Ballroom E**
AIAA-94-3408 **Ada, Autocoding, and Object-Based Software Design—An Application to Real Time Flight Controls Simulation** B. Green, *Lockheed Aeronautical Systems Co., Marietta, GA*	**AIAA-94-3409** **Boeing 777 Systems Integration Lab Simulation System and Software Architecture** R. Kircher, *The Boeing Co., Seattle, WA*	**AIAA-94-3410** **I/O Systems Facilitate the Development and Certification of New Commercial Airplanes** J. Ortiz, *Boeing Commercial Airplane Co., Seattle, WA*	**AIAA-94-3411** **Increasing the Flexibility and Reducing the Cost of Flight Simulators** E. Theunissen, *Delft Univ. of Technology, Delft, The Netherlands*	**AIAA-94-3412** **Application of Graphical Software Generation Technology in Simulation of All-Digital Tranport Aircraft** S. Fookes, *Boeing Commercial Airplane Co., Seattle, WA*	**AIAA-94-3413** **Simulation of a Redundant Digital Asynchronous Flight Control System in a Multi-Process Environment** B. Parris and L. Rader, *Boeing Commercial Airplane Co., Seattle, WA*		

Monday Afternoon / August 1, 1994

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
Session 29-FST-4 Simulation Applications I Chaired by: S. ADVANI, Delft University of Technology, Delft, The Netherlands							**Room 6**
AIAA-94-3414 **Development of the Dutch National Simulation Facility** H. Offerman, *National Aerospace Lab, Amsterdam, The Netherlands*	**AIAA-94-3415** **Modern Flight Simulators for Research Applications** H. Bernard, *CityLine Simulator and Training GmbH, Berlin, Germany;* G. Huettig, *Berlin Univ. of Technology, Berlin, Germany*	**AIAA-94-3416** **The "Smart Software-Simple Hardware" Concept for Maximum Flexibility in Research Flight Simulation** J. Hoekstra, *National Aerospace Lab, Amsterdam, The Netherlands*	**AIAA-94-3417** **Advanced Missile Concept Evaluation Toolset** C. Ewing, *Wright Lab, Eglin AFB, FL*	**AIAA-94-3418** **The Influence of Platform Inertia on Simulator Motion System Performance** S. Advani and R. Verbeek, *Delft Univ. of Technology, Delft, The Netherlands*			
Session 30-ASD-4 OD and Navigation Chaired by: R. SCHINNERER, Loral Space Systems, Colorado Springs, CO							**Room 10**
AIAA-94-3722 **Analytical Expressions for Orbit Covariance due to Gravity Errors** J. Wright, *Applied Technology Assoc., Inc., Exton, PA*	**AIAA-94-3723** **On the Numerical Computation of Multi-Dimensional Integrals Appearing in the Orbit Error Propagations** K. Rao, *INEP, São Jose dos Campos, Brazil*	**AIAA-94-3724** **A Comparison of Satellite Aerodynamic Models Using ERS-1 Satellite Laser Ranging Data** I. Harrison and G. Swinerd, *Univ. of Southampton, Southampton, UK;* G. Appleby, *Cambridge, UK*	**AIAA-94-3725** **Single Pass Orbit Adjustment for the NOAA Polar Orbiting Satellites Using DCLS Doppler Measurements** S. Smith and J. Lundberg, *Univ. of Texas at Austin, Austin, TX*	**AIAA-94-3726** **The Radarsat Flight Dynamics System: An Extensible, Portable, Workstation-Based Mission Support System** P. Cefola, R. Proube, R. Metzinger, and M. Cohen, *C. S. Draper Lab, Cambridge, MA*	**AIAA-94-3727** **Possible Correlation Between Geomagnetic Planetary Index and Satellite Catalog Accuracy** T. Eller and P. Snow, *Kaman Sciences, Colorado Springs, CO*		
Session 31-ASD-5 Orbit/Attitude Guidance and Control I Chaired by: R. JACOBSON, Jet Propulsion Laboratory, Pasadena, CA							**Room 11**
AIAA-94-3728 **A New Approach to Halo Orbit Determination and Control** D. Cielaszyk and B. Wie, *Arizona State Univ., Tempe, AZ*	**AIAA-94-3729** **Halo Orbit Determination and Control for the Eccentric-Restricted Three Body Problem** B. Wie, *Arizona State Univ., Tempe, AZ*	**AIAA-94-3730** **Propellant Slosh Models for the Cassini Spacecraft** P. Enright and E. Wang, *JPL, Pasadena, CA*	**AIAA-94-3731** **Pole Placement in Time Dependent Linear Systems** W. Wiesel, *Air Force Inst. of Technology, Wright-Patterson AFB, OH*	**AIAA-94-3732** **Optimal Guidance and Nonlinear Estimation for Interception of Accelerating Targets** M. Hough, *Textron Defense Systems, Wilmington, MA*	**AIAA-94-3733** **Flight Envelopes for Hypersonic Aero-Rendezvous with Large Initial Heading** Z-C. Hong and S-J. Chen, *National Central Univ., Chungli, Taiwan, ROC;* J-S. Chem, *Lungtan, Taiwan, ROC*		
Session 32-ASD-6 Tethered Satellite Systems I Chaired by: P. BAINUM, Howard University, Washington, DC							**Room 12**
AIAA-94-3734 **Technical Note on the Observations of TSS System Dynamics (STS-46)** G. Bianchini and F. Angrilli, *Univ. of Padova, Padova, Italy;* G. Fanti, *Univ. of Parma, Italy*	**AIAA-94-3735** **Nonlinear Vibrations in TSS-1** S. Bergamaschi, P. Zanetti, and C. Zottarel, *Univ. of Padova, Padova, Italy*	**AIAA-94-3736** **Control of Tethered Satellite Systems Using Thruster and Offset Strategies** V. Modi and S. Pradhan, *Univ. of British Columbia, Vancouver, British Columbia, Canada;* A. Misra, *McGill Univ., Montreal, Quebec, Canada*	**AIAA-94-3737** **A Direct Method to Generate the Minimum-Order Equations of Motion for Systems with Artificially Imposed Constraints** A. Misra and M. Sadigh, *McGill Univ., Montreal, Quebec, Canada*	**AIAA-94-3738** **Retrieval/Deployment of an Orbiting Tethered Antenna/Reflector System Test Scale Model** P. Bainum and Z. Li, *Howard Univ., Washington, DC*	**AIAA-94-3739** **The Analysis on the Motion Stability of a Tethered Satellite System** C. Naigang, L. Dun, Y. Daming, and L. Yuhua, *Harbin Inst. of Technology, Harbin, China*		
Offsite Event	**7:00 pm – 10:00 pm**						

Tuesday Morning / August 2, 1994

Plenary Session **Low-Temperature Inertial Instruments (8:00 am – 9:00 am)** **Estrella Theatre**
Daniel B. DeBra, Ed Wells Professor of Aeronautics and Astronautics, Stanford University, Stanford, CA

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 33-GNC-15 Flexible Structure Identification and Control Chaired by: L. KEEL, Tennessee State University, Nashville, TN							**Ballroom A**
AIAA-94-3621 **Structural Mode Identification of Galileo Spacecraft from Flight Data** C-H. Ih, S-R. Lin, E. Wong, and G. Macala, *JPL, Pasadena, CA*	**AIAA-94-3622** **Structural System Identification: A Study of Different Algorithms** K. Liu and D. Miller, *MIT, Cambridge, MA*	**AIAA-94-3623** **Practical Structural Mode Identification** M. Barrett, *Honeywell Technology Center, Minneapolis, MN*	**AIAA-94-3624** **Near-Minimum-Time Maneuvers of Large Structures: Theory and Experiments** S. Vadali, M. Carter, T. Singh, and N. Abhyankar, *Texas A&M Univ., College Station, TX*	**AIAA-94-3625** **Application of Constraint Dynamics for Spacecraft Maneuver** H. Bank and B. Agrawal, *Naval Postgraduate School, Monterey, CA*	**AIAA-94-3626** **Optimal Acceleration Profile for Flexible Space Structures** H. Fuji, H. Hatakenaka, and M. Nakagawa, *Tokyo Metropolitan Inst. of Technology, Tokyo, Japan*		
Session 34-GNC-16 Further Control Applications Chaired by: D. GROLL, McDonnell Douglas Aerospace, St. Louis, MO							**Ballroom F**
AIAA-94-3627 **Standard Optimal Pilot Models** M. Anderson, *VPI, Blacksburg, VA*	**AIAA-94-3628** **The Effect of Actuator Non-Linearities on Aero-Servo-Elasticity** R. Taylor, R. Pratt, and B. Caldwell, *Loughborough Univ., Loughborough, UK*	**AIAA-94-3629** **Further Flight-Dynamic-and-Control Analysis of Elastic Hypersonic Vehicles** F. Chavez and D. Schmidt, *Univ. of Maryland, College Park, MD*	**AIAA-94-3630** **Flight Control Law Synthesis for a Flexible Aircraft** F. Kubica, *Aerospatiale, Toulouse, France*	**AIAA-94-3631** **Highly Augmented Control Mode Concepts and Simulation Evaluation for STOVL Aircraft** D. Bodden and J. Vimig, *Lockheed Fort Worth Co., Ft. Worth, TX*			
Session 35-GNC-17 Guidance Problems: Theory and Navigation Chaired by: B. Sacleux, ONERA, Chatillo, FRANCE							**Ballroom G**
AIAA-94-3632 **A General Guidance Law: Application to a Launch Vehicle** P. Lu and M. Khan, *Iowa State Univ., Ames, IA*	**AIAA-94-3633** **Mixed Strategy Guidance in Future Ship Defense** Y. Lipman and J. Shinar, *Technion, Haifa, Israel*	**AIAA-94-3634** **Altitude-Path Angle Control During Aerospace Plane Ascent** J-P. Kremer and K. Mease, *Univ. of California at Irvine, Irvine, CA*	**AIAA-94-3635** **Near-Optimal Propulsion System Operation for Air-Breathing Launch Vehicles** M. Ardema, J. Bowles, and T. Whittaker, *NASA Ames, Moffett Field, CA*	**AIAA-94-3636** **Design and Test of a Parallel Trajectory Optimization Algorithm** K. Park and M. Psiaki, *Cornell Univ., Ithaca, NY*			
Session 36-GNC-18 Flexible Structure Control III Chaired by: S. VADALI, Texas A&M University, College Station, TX							**Room 7**
AIAA-94-3637 **Tests of Piezoelectric Proof-Mass Actuators for Disturbance Rejection** C. Won, *Lockheed Engineering & Sciences Co., Hampton, VA*	**AIAA-94-3638** **Optimal Sensor Placement and Active Vibration Suppression of Large Flexible Space Structures** E. Tongco and D. Meldrum, *Univ. of Washington, Seattle, WA*	**AIAA-94-3639** **Multiplexed Control for Massively Actuated Structures** D. Smart and L. Peterson, *Univ. of Colorado at Boulder, Boulder, CO*	**AIAA-94-3640** **Discrete Time State Space Models Based on Response Data Over Finite Time Interval** M. Baker and D. Mingori, *Univ. of California at Los Angeles, Los Angeles, CA*	**AIAA-94-3641** **Control of Flexible Structures Using Model Following Control Scheme** Y. Kim and C. Yang, *Seoul National Univ., Seoul, Korea*	**AIAA-94-3642** **LQG Control of Flexible Structures Using Piezoelements** A. Youseli-Koma, J. Sasiadek, and G. Vukovich, *Carleton Univ., Ottawa, Ontario, Canada*		

Tuesday Morning / August 2, 1994

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 37-GNC-19 Neural Network Application in Flight Control Chaired by: A. CALISE, Georgia Institute of Technology, Atlanta, GA							**Room 8**
AIAA-94-3643 **On-Line Learning Non-Linear Direct Neuro Controllers for Restructurable Control Systems** M. Napolitano, V. Casdorph, S. Naylor, and C. Neppach, *West Virginia Univ., Morgantown, WV*	**AIAA-94-3644** **On the Application of Neural Network Computing to the Constrained Flight Control Allocation Problem** R. Grogan, W. Durham, and R. Krishnan, *VPI, Blacksburg, VA*	**AIAA-94-3645** **Comparison of Neural Network Based, Fuzzy Logic Based and Numerical Nonlinear Inverse Flight Controls** C. Huang, J. Tylock, S. Engel, and J. Whitson, *Grumman Corp., Bethpage, NY*	**AIAA-94-3646** **Nonlinear Flight Control using Neural Networks** B. Kim and A. Calise, *Georgia Inst. of Technology, Atlanta, GA*				
Session 38-GNC-20 Spacecraft Position Estimation, Station Keeping, and Maneuvering Chaired by: T. ALT, McDonnell Douglas Aerospace, Huntington Beach, CA							**Room 9**
AIAA-94-3647 **Optimal Control Solutions for an Aeroassisted Orbit Transfer Problem with a Heating Rate Limit** H. Seywald, *Analytical Mechanics Associates, Inc., Hampton, VA*	**AIAA-94-3648** **In-Orbit Demonstration of Unmanned Automatic Rendez-vous and Docking System by Japanese Engineering Test Satellite ETS-VII** I. Kawano, *National Space Development Agency of Japan, Tokyo, Japan*	**AIAA-94-3649** **Fuel Optimal Station-Keeping Via Differential Inclusions** R. Kumar, *Analytical Mechanics Assoc., Hampton, VA*	**AIAA-94-3650** **Numerical Computation of Fuel-Optimal Low- and Medium-Thrust Orbit Transfers in Large Numbers of Burns** C-H. Chuang, *Georgia Inst. of Technology, Atlanta, GA*	**AIAA-94-3651** **The Lunar Penetrators Position Estimate that the Spacecraft was Utilized for . . .** T. Ichikawa, *Inst. of Space & Astronautical Science, Kanagawa, Japan*			
Session 39-AFM-9 Vortical/Vortex Aerodynamics Chaired by: S. YING, Iowa State University, Ames, IA							**Room 1**
AIAA-94-3483 **Semi-Empirical Analysis of Vortex Breakdown with Aerodynamic & Buffet Effects** C. Dixon, *Consulting Aviation Services, Kennesaw, GA*	**AIAA-94-3484** **Dissimilar Aircraft Flying in Close Proximity** D. Porter and R. Howard, *Naval Postgraduate School, Monterey, CA*	**AIAA-94-3485** **Asymmetric Vortex Wake on the Ogive Slender Body** N. Chen and Z. Wang, *Beijing Univ. of Aeronautics & Astronautics, Beijing, China*	**AIAA-94-3486** **Simulation of Vertical Tail Buffet in Internal Vortex Breakdown Flows** O. Kandil and M. Flanagan, *Old Dominion Univ., Norfolk, VA*	**AIAA-94-3487** **Vortex Breakdown Control by Delta Wing Geometry** M. Lowson and A Riley, *Univ. of Bristol, Bristol, UK*			
Session 40-AFM-10 Evaluation of Aircraft Flying Qualities Chaired by: R. LIVNEH, Arizona State University, Tempe, AZ							**Room 2**
AIAA-94-3488 **Helicopter Roll-Pitch Coupling Feedback Model and Comparisons with Handling Qualities Flight Test Data** S. Mouritsen, *DLR, Braunschweig, Germany*	**AIAA-94-3489** **Comparison of Results from an In-Flight and a Ground-Based Simulation of Longitudinal Flying Qualities for Augmented, Large Transports in Approach and Landing** J. Preston, K. Rossitto, and J. Hodgkinson, *McDonnell Douglas Aerospace, Long Beach, CA*	**AIAA-94-3490** **An Aerial Slalom Course for Flying Qualities Evaluation** T. Cord, *Wright Lab, Wright-Patterson AFB, OH*	**AIAA-94-3491** **Mission-Specific Flying Qualities Criteria** M. Page and D. Gillette, *McDonnell Douglas Aerospace, Long Beach, CA*	**AIAA-94-3492** **A Look at Handling Qualities of High Performance Hang Gliders** S. Anderson and R. Ormiston, *NASA Ames, Moffett Field, CA*			

Tuesday Morning / August 2, 1994

<table>
<tr><th>9:00</th><th>9:30</th><th>10:00</th><th>10:30</th><th>11:00</th><th>11:30</th><th>12:00</th><th>12:30</th></tr>
<tr><td colspan="7"><b>Session 41-AFM-11 Aircraft Dynamics II</b>
Chaired by: C. BLACKLOCK JR., Lockheed Fort Worth Company, Forth Worth, TX</td><td><b>Room 3</b></td></tr>
<tr>
<td><b>AIAA-94-3493</b>
<b>Simplified Nonlinear Indicial Response Models of a Rolling 65 Degree Delta Wing</b>
A. Hsia and J. Jenkins, <i>Wright Lab, Wright-Patterson AFB, OH</i></td>
<td><b>AIAA-94-3494</b>
<b>Estimation of Aircraft Models in Cobra Maneuver through Dynamic Inversion</b>
Y. Li, F. Ge, and C. Lan, <i>Univ. of Kansas, Lawrence, KS</i></td>
<td><b>AIAA-94-3495</b>
<b>Time Optimal Sidestep Maneuvers of Aircraft</b>
Y. Fan, F. Lutze, and E. Cliff, <i>VPI, Blacksburg, VA</i></td>
<td><b>AIAA-94-3496</b>
<b>Aerodynamic Characteristics of Aircraft Under Side Gust</b>
I. Olwi, <i>King Abdulaziz Univ., Jeddah, Saudi Arabia</i></td>
<td><b>AIAA-94-3497</b>
<b>Effects of High-G Loads on Dynamics of Closed-Loop Pilot-Aircraft Systems</b>
Y. Yashin, A. Predtechensky, G. Severin, and A. Barer, <i>Central Aerohydrodynamic Inst., Moscow, Russia</i></td>
<td></td><td></td><td></td>
</tr>
<tr><td colspan="7"><b>Session 42-AFM-12 Missile Aerodynamics</b>
Chaired by: T. SHIVANANDA, TRW Space and Defense, San Bernardino, CA</td><td><b>Room 5</b></td></tr>
<tr>
<td><b>AIAA-94-3498</b>
<b>Aerodynamic Test and Analysis of a Slender, Grooved-Body, Generic Missile Configuration</b>
W. Clay Howerton, <i>Wright Lab, Eglin AFB, FL;</i> W. Hathaway, <i>Arrow Tech Associates, South Burlington, VT</i></td>
<td><b>AIAA-94-3499</b>
<b>Wind Tunnel Measurements of Wrap Around Fins at Mach 2.06</b>
G. Abate, <i>Wright Lab, Eglin AFB, FL;</i> C. Berner, <i>French-German Research Inst., Saint Louis, France</i></td>
<td><b>AIAA-94-3500</b>
<b>Unsteady Aerodynamics of a Transient Pitching Missile Body from 0 to 25 Degrees</b>
T. Hseih and A. Wardlaw Jr., <i>Naval Warfare Center, Silver Spring, MD</i></td>
<td><b>AIAA-94-3501</b>
<b>HYFLEXS Computational Fluid Dynamics Analysis</b>
Y. Yamamoto, <i>National Aerospace Lab, Tokyo, Japan;</i> M. Yoshioka, <i>Fujitsu Ltd., Tokyo, Japan</i></td>
<td><b>AIAA-94-3502</b>
<b>Computation of Turbulent Flow Over Finned Projectiles</b>
P. Kaurinkoski, E. Salminen, and T. Siikonen, <i>Helsinki Univ. of Technology, Espoo, Finland</i></td>
<td></td><td></td><td></td>
</tr>
<tr><td colspan="7"><b>Session 43-FST-5 Simulation Applications II</b>
Chaired by: K. ZWAANENBURG, Applied Dynamics International, Ann Arbor, MI</td><td><b>Ballroom E</b></td></tr>
<tr>
<td><b>AIAA-94-3419</b>
<b>A Part Task Simulator for Advanced Automation and Communications Research</b>
G. Pisanich, E. Lee, and L. Beck, <i>Sterling Software, Moffett Field, CA</i></td>
<td><b>AIAA-94-3420</b>
<b>Manned Flight Simulator and the Impact on Navy Weapon System Acquisition</b>
R. Mills, R. Burton, and C. Miller, <i>Naval Air Warfare Center, Patuxent River, MD</i></td>
<td><b>AIAA-94-3421</b>
<b>A Feasibility Study of the Use of Axisymmetric Thrust Vectoring for Enhanced In-Flight Simulation</b>
M. Jones, <i>Wright Lab, Wright-Patterson AFB, OH;</i> D. Andrisani II, <i>Purdue Univ., West Lafayette, IN</i></td>
<td><b>AIAA-94-3422</b>
<b>Flight Test Data Modeling and Compression by Fuzzy Extracted Topology Technique</b>
D. Hensley, <i>Boeing Commercial Airplanes, Kent, WA</i></td>
<td><b>AIAA-94-3423</b>
<b>Operational Prototype for an Instructor/Operation Station</b>
R. Fulton, <i>Enhanced Technologies, Houston, TX;</i> T. Diegelman, <i>NASA Johnson, Houston, TX;</i> D. Webster, <i>CAE-Link Corp., Houston, TX</i></td>
<td><b>AIAA-94-3424</b>
<b>A Facility for Simulating Space Station On-Orbit Procedures</b>
A. Hajare, D. Wick, and N. Shehad, <i>Enhanced Technologies, Houston, TX</i></td>
<td></td><td></td>
</tr>
</table>

Tuesday Morning / August 2, 1994

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 44-FST-6 Networking and Modeling Structure Chaired by: D. SCHAB, Naval Air Warfare Center, Orlando, FL							**Room 6**
AIAA-94-3425 **Study on the Network Load in Distributed Interactive Simulation** K-C. Lin, *Institute for Simulation and Training, Orlando, FL;* D. Schab, *Naval Air Warfare Center, Orlando, FL*	**AIAA-94-3426** **Somnambulistic Reckoning for Distributed Interactive Simulations** D. Bernard and K. Forsstrom, *Northrop, Pico Rivera, CA*	**AIAA-94-3427** **Shared Memory Networks: A Key Enabling Technology for Distributed Real-Time Simulation** G. Valentino, *SYSTRAN Corp., Dayton, OH*	**AIAA-94-3428** **Extrapolation of Airplane Flight** A. Katz and K. Graham, *Univ. of Alabama, Tuscaloosa, AL*	**AIAA-94-3429** **The Generic Simulation Executive at Manned Flight Simulator** J. Nichols, *Naval Air Warfare Center, Patuxent River, MD*	**AIAA-94-3430** **A Recommended Process for Developing Simulation Modeling Standards** B. Hildreth, *SAIC, Lexington Park, MD*		
Session 45-ASD-7 Invited Astrodynamics Plenary Chaired by: P. CEFOLA, C. S. Draper Laboratory, Cambridge, MA, and T. ELLER, Kaman Sciences Corporation, Colorado Springs, CO							**Estrella Theatre**
Panelists TBD							

Awards Luncheon
12:00 – 2:00 pm
Ballrooms B, C, and D
Story Musgrave (invited)
Astronaut
NASA Johnson Space Flight Center
Houston, TX
On the Hubble Repairs

Mechanics and Control of Flight Award
This award is presented for an outstanding recent technical or scientific contribution by an individual in the mechanics, guidance, or control of flight in space or the atmosphere.

De Florez Training Award for Flight Simulation
This award is named in honor of the late Admiral de Florez and is presented for an outstanding achievement in the application of flight simulation to aerospace training research and development.

Tuesday Afternoon / August 2, 1994

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
Session 46-GNC-21 Space Robotics and Automation Chaired by: J. SASIADEK, Carleton University, Ottawa, Ontario, Canada							**Ballroom A**
AIAA-94-3652 **Free-Flying Robot Equipped with Thruster Jets** J. Sasiadek, I. Duleba, and G. Vukovich, *Carleton Univ., Ottawa, Ontario, Canada*	AIAA-94-3653 **Cooperative Control of Multiple Space Manipulators** G. Yale and B. Agrawal, *Naval Postgraduate School, Monterey, CA*	AIAA-94-3654 **Nonlinear Control of Space Manipulators with Model Uncertainty** M. Mittal, C. Chuang, and J. Juang, *Georgia Inst. of Technology, Atlanta, GA*	AIAA-94-3655 **Optimal Rotational Maneuvers of Spacecraft Using Manipulator Arms** S. Krishnan and S. Vadali, *Texas A&M Univ., College Station, TX*	AIAA-94-3656 **Manipulation Variable Feedback Control of Flexible Manipulators by Using Virtual Rigids Manipulator Concept** K. Senda and Y. Murotsu, *Univ. of Osaka Prefecture, Osaka, Japan*	AIAA-94-3657 **Dynamics and Control of Wind Driven Manipulators** J. Magill, N. Komerath, and J. Dorsey, *Georgia Inst. of Technology, Atlanta, GA*		
Session 47-GNC-22 H ∞ Control Chaired by: D. PRICE, NASA Langley Research Center, Hampton, VA							**Ballroom F**
AIAA-94-3658 **Reduced Order Mixed H2/ H∞ Optimization with Multiple Hinf Constraints** J. Ullauri, D. Walker, and D. Ridgely, *Air Force Inst. of Technology, Wright-Patterson AFB, OH*	AIAA-94-3659 **Flight Controller Design Using Mixed H2/H∞ with a Singular Hinf Constraint** J. Luke, D. Ridgely, and D. Walker, *Air Force Inst. of Technology, Wright-Patterson AFB, OH*	AIAA-94-3660 **A Differential Game Approach to the Mixed H2/ H∞ Problem** G. Swerdiuk and A. Calise, *Georgia Inst. of Technology, Atlanta, GA*	AIAA-94-3661 **Homotopy Algorithms for Fixed Order H2 and H∞ Design** M. Whorton, H. Buschek, and A. Calise, *NASA Marshall, Huntsville, AL*	AIAA-94-3662 **Fixed Order Robust Control Design for Hypersonic Vehicles** H. Buschek and A. Calise, *Georgia Inst. of Technology, Atlanta, GA*			
Session 48-GNC-23 Control Theory II Chaired by: M. ANDERSON, Virginia Polytechnic Institute and State University, Blacksburg, VA							**Ballroom G**
AIAA-94-3663 **Attitude Filtering Method for Highly Maneuverable Spacecraft** B. Sacleux and F. Damongeot, *ONERA, Chatillon, France*	AIAA-94-3664 **Nonlinear State Feedback Control of Space Vehicle Model Using a Real-Time Optimization Technique** T. Ohtsuka and H. Fujii, *Tokyo Metropolitan Inst. of Technology, Tokyo, Japan*	AIAA-94-3665 **Extra-Insensitive Input Shapers for Controlling Flexible Spacecraft** W. Singhose, S. Derenzinski, and N. Singer, *Convolve, Inc., Armonk, NY*	AIAA-94-3666 **The Minimum Control Authority for Flexible Structures** E. Tongco, D. Meldrum, and P. Wiktor, *Univ. of Washington, Seattle, WA*	AIAA-94-3667 **Nonlinear Control via the Algebraic Equations of Motion Applied to an Aerodynamic Body** G. Fuerst and H. Oz, *Ohio State Univ., Columbus, OH*	AIAA-94-3696 **Use of a Laboratory Tilt Rotor Aircraft to Develop Practical Knowledge of Flying Qualities Issues** D. Doman and D. Andriasani II, *Wright Laboratory, Wright-Patterson AFB, OH*		
Session 49-GNC-24 Pilot-Induced Oscillations: Prediction and Alleviation Chaired by: TBD							**Room 7**
AIAA-94-3668 **Summary of an AGARD Workshop on Pilot-Induced oscillations** K. McKay, *British Aerospace Defence, Preston, UK*	AIAA-94-3669 **Predicting and Validating Fully-Developed PIO** R. Smith, *High Plains Engineering, Tehachapi, CA*	AIAA-94-3670 **The Measurement and Prediction of Pilot-in-the-Loop Oscillations** D. Mitchell, R. Hoh, B. Aponso, and D. Klyde, *Hoh Aeronautics, Inc., Lomita, CA*	AIAA-94-3671 **Experience with the R Smith PIO Criterion in the F-15 STOL/Maneuver Technology Demonstrator** D. Moorhouse, *USAF, Wright-Patterson AFB, OH*	AIAA-94-3672 **Multivariable Design to Directly Satisfy Flying Qualities** E. Rynaski, *EGR Associates, Buffalo, NY*	AIAA-94-3673 **An Alternate Control Scheme to Alleviate Aircraft-Pilot Coupling** R A'Harrah, *NASA Headquarters, Washington, DC*		

Tuesday Afternoon / August 2, 1994

Session 50-GNC-25 Integrated Navigation Systems

Room 8

Chaired by: J. DOWDLE, C. S. Draper Laboratory, Cambridge, MA

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
AIAA-94-3674 **Synthetic Vision and Precision Navigation for Aircraft Taxi Guidance in Low Visibility** G. Sachs, H. Moller, K. Dobler, G. Schanzer, and K. Mohlenkamp, *Technische Univ. Munchen, Munchen, Germany*	AIAA-94-3675 **Consider-Filter for Attitude Determination** I. Bar-Itzhack and J. Nathan, *Technion, Haifa, Israel*	AIAA-94-3676 **Attitude Determination of a Rotating Body Using GPS and INS Data** A. Soltz, *C. S. Draper Lab, Cambridge, MA*	AIAA-94-3677 **Autonomous Navigation with Passive Imaging Sensors** C-F. Lin and A. Politopoulos, *American GNC Corp., Canoga Park, CA;* D. Elliott, *USAF, Norton AFB, CA*	AIAA-94-3678 **GPS Aided Passive Imaging Navigation** C-F. Lin and R. Da, *American GNC Corp., Canoga Park, CA;* D. Elliott, *USAF, Norton AFB, CA*			

Session 51-GNC-26 Control of Highly Maneuverable Aircraft

Room 9

Chaired by: J. HODGKINSON, McDonnell Douglas Aerospace, Long Beach, CA

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
AIAA-94-3679 **Parameter Robust Game Theoretic Synthesis for the F-18 HARV** C. Dillon and J. Speyer, *Univ. of California at Los Angeles, Los Angeles, CA*	AIAA-94-3680 **Adaptive and Neural Control of Wing-Rock Motion of Slender Delta Wing** S. Singh, W. Yim, and W. Wells, *Univ. of Nevada at Las Vegas, Las Vegas, NV*	AIAA-94-3681 **Active Vortex Flow Control for VISTA F-16 Envelope Expansion** R. Adams, J. Buffington, and S. Banda, *USAF, Wright-Patterson AFB, OH*	AIAA-94-3682 **Design of a Flight Control System for a Highly Maneuverable Aircraft Using Robust Dynamic Inversion** J. Reiner, G. Balas, and W. Garrard, *Univ. of Minnesota, Minneapolis, MN*	AIAA-94-3683 **Control Stick Logic in High Angle-of-Attack Maneuvering** W. Durham, *VPI, Blacksburg, VA*	AIAA-94-3684 **A New Measure of Departure Resistance Using Structured Singular Values** B. York and M. Anderson, *Systems Control Technology, Inc., Lexington Park, MD*		

Session 52-AFM-13 High Angle-of-Attack Flight

Room 1

Chaired by: F. GARRETT JR., Arizona State University, Tempe, AZ

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
AIAA-94-3504 **Parameter Estimation for the NASA F/A-18 HARV at High Angles of Attack** M. Napolitano, A. Paris, and J. Spagnuolo, *West Virginia Univ., Morgantown, WV;* A. Bowers, *NASA Dryden, Edwards, CA*	AIAA-94-3506 **Development of Flying Qualities Criteria for 60 Degree Angle of Attack** D. Wilson, K. Citurs, and J. Davidson, *McDonnell Douglas Aerospace, St. Louis, MO*	AIAA-94-3507 **Leading-Edge Vortex Behavior on a 65 Degree Delta Wing Oscillating in Roll** X. Huang and E. Hanff, *National Research Council, Ottawa, Ontario, Canada;* J. Jenkins and G. Addington, *Wright Lab, Wright-Patterson AFB, OH*					

Tuesday Afternoon / August 2, 1994

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
Session 53-AFM-14 Aircraft Simulation and Flight Test Chaired by: W. BLAKE, Wright Laboratory, Wright-Patterson AFB, OH							**Room 2**
AIAA-94-3508 **Simulator and Flight Tests on Aerospace Plane Long-Period Control and Flying Qualities** G. Sachs, R. Stich, and A. Knoll, *Technische Univ. Munchen, Munchen, Germany;* T. Cox, *NASA Dryden, Edwards, CA*	AIAA-94-3509 **Ship Airwake Effects of Helicopter Rotor Aerodynamic Loads** H. Zhang, J. Prasad, and D. Marvis, *Georgia Inst of Technology, Atlanta, GA*	AIAA-94-3510 **Results of a Simulator Investigation into the Effects of the Instantaneous Center of Rotation on Approach and Landing Flying Qualities** K. Rossitto and J. Preston, *McDonnell Douglas Aerospace, Long Beach, CA*	AIAA-94-3511 **A Near Realtime Approach to Statistical Flight Test** B. Jones, *Air Force Inst. of Technology, Wright-Patterson AFB, OH*	AIAA-94-3512 **Evaluation Maneuver and Guideline Development for High-Alpha Control Law Design Using Piloted Simulation** K. Hoffler, *ViGYAN, Inc., Hampton, VA;* P. Brown, M. Phillips, R. Rivers, J. Davidson Jr., F. Lallman, P. Murphy, and A. Ostroff, *NASA Langley, Hampton, VA*			
Session 54-AFM-15 Stability and Control II Chaired by: J. KALVISTE, Northrop Corporation, Pico Rivera, CA							**Room 3**
AIAA-94-3459 **Flight-Estimated Spoiler Aerodynamics of the F-111C Aircraft** R. Stuckey, *Univ. of Sydney, Sydney, Australia*	AIAA-94-3514 **Flight Control Design Directly from Mil-F-8785(C)** E. Rynaski, *EGR Associates, Buffalo, NY*	AIAA-94-3515 **On the Use of Controls for Subsonic Transport Performance Improvement: Overview and Future Directions** G. Gilyard and M. España, *NASA Ames, Edwards, CA*	AIAA-94-3516 **Functional Control Law Design Using Exact Nonlinear Dynamic Inversion** P. Smith, *Defence Research Agency, Bedford, UK*	AIAA-94-3517 **Nonlinear Decoupling Control of Aircraft Motion** Z. Zhiqiang, *Northwestern Polytechnical Univ., Xian, China*	AIAA-94-3518 **Lateral Stability Augmentation Using Decentralized Control** J. Jang, J. Kim, and C. Park, *Inha Univ., Inchon, Korea*		
Session 55-FST-7 Rotorcraft Simulation Chaired by: M. FERRANTI, Rail Company, Stratford, CT							**Ballroom E**
AIAA-94-3431 **Development of a Deployable AH-1W Trainer** C. Miller and D. Perdue, *Naval Air Warfare Center, Patuxent River, MD*	AIAA-94-3432 **Vertical Axis Rotational Cues in Hovering Flight Simulation** J. Schroeder and W. Johnson, *NASA Ames, Moffett Field, CA*	AIAA-94-3433 **Helicopter In-Flight Simulation via a VSS: Development and Use in Test Pilot Training** R. Miller, *U.S. Naval Test Pilot School, Patuxent River, MD;* L. Khinoo, *Veda, Inc., Lexington Park, MD*	AIAA-94-3434 **Modeling Techniques Improving Fidelity of Aircraft/Ship Dynamic Interface Simulation** J. Funk Jr., and C. Beck, *Naval Air Warfare Center, Patuxent River, MD;* J. Blackwell, *DSTO Aeronautical Research Lab, Fishermens Bend, Australia*	AIAA-94-3435 **Helicopter Training for Naval Training** D. Ledar, *Thomson-CSF, Osny, France*	AIAA-94-3436 **A Blade Element Rotor Model in a Low Cost Helicopter Simulator** A. Katz and K. Graham, *Univ. of Alabama, Tuscaloosa, AL*		

Tuesday Afternoon / August 2, 1994

2:00	2:30	3:00	3:30	4:00	4:30	5:00	5:30
Session 56-ASD-8 Orbit Analysis II Chaired by: J. LUNDBERG, University of Texas, Austin, TX **Room 10**							
AIAA-94-3740 **Calculating Collisions Rates, Intersection Angles and Relative Velocities for Orbital Debris Application** R. Malder and R. Culp, *Univ. of Colorado at Boulder, Boulder, CO*	**AIAA-94-3741** **Assessment of the Short-Term Collision Hazard Resulting from an On-Orbit Fragmentation Event: Polar Platform Case Study** S. Barrows and G. Swinerd, *Univ. of Southampton, Southampton, UK;* R. Crowther, *Farnborough, UK*	**AIAA-94-3742** **Improvements in Venus Gravity Field Determination Using Ridge-Type Estimation Methods** D. Cicci, *Auburn Univ., Auburn, AL;* M. Johnson, *Lockheed Missiles and Space Co., Sunnyvale, CA*	**AIAA-94-3743** **Solar Panel Ejection on Orbit: Searching for Stability and Identifying Chaotic Regions** W. Kelly and M. Brookover, *Lockheed Engineering and Sciences Co., Houston, TX;* J. Wade, *NASA Johnson, Houston, TX*	**AIAA-94-3744** **A Compact Finite Burn Targeting and Reconstruction Algorithm** S. Hast, D. Oltrogge, and M. Hart, *The Aerospace Corp., Los Angeles, CA*	**AIAA-94-3745** **Transfer Orbits in the Restricted Problem** A. Prado, *INPE-DGM, São Jose dos Campos, Brazil;* R. Broucke, *Austin, TX*		
Session 57-ASD-9 Catchup OAGC II and Tethered Satellite Systems II Chaired by: D. SPENCER, Phillips Laboratory, Kirtland AFB, NM **Room 11**							
AIAA-94-3746 **The Calculation of the Orbit Elements of a Tethered Satellite After Release** C. Naigang, L. Dun, L. Yuhua, and Q. Naiming, *Harbin Inst. of Technology, Harbin, China*	**AIAA-94-3747** **Analysis of the Optimal Mass Problem for Aerobraking Tethers** S. Tragesser, J. Longuski, J. Puig-Suari, and J. Mechalas, *Purdue Univ., West Lafayette, IN*	**AIAA-94-3748** **Flight Control, Dynamics and Structural Interaction on the Space Shuttle Hubble Space Telescope Servicing Mission** L. Sackett and C. Kirchway, *C. S. Draper Lab, Cambridge, MA*	**AIAA-94-3749** **Orbital Maneuvers Via Feedback Linearization and Bang-Bang Control** J. Cochran Jr., *Auburn Univ., Auburn Univ., AL;* S. Lee, *Taejun, Korea*	**AIAA-94-3750** **Exo-Atmospheric Intercept Using Lagrange's Prediction Equations** D. Zes, *Orbital Sciences Corp., Mesa, AZ*	**AIAA-94-3751** **Ground Experiments of a Hanging Spinning Tethered Body** G. Tyc, V. Modi, S. Pradhan, R. Han, and A. Misra, *Bristol Aerospace Ltd., Winnipeg, Manitoba, Canada*		
Session 58-ASD-10 Dynamics and Control of Flexible Structures Chaired by: V. MODI, University of British Columbia, Vancouver, British Columbia, Canada **Room 12**							
AIAA-94-3752 **Slewing and Vibration Control of a Nonlinear Flexible Shuttle Antenna** A. Banerjee and Y. Sahinkaya, *Lockheed Palo Alto Research Lab, Palo Alto, CA*	**AIAA-94-3753** **Spacecraft Attitude Control using Slide Mode Control with Uncontrollable Flexible Structural Dynamics** J. Kim, *Inha Univ., Inchon, Korea;* T. Dwyer, *Univ. of Illinois, Chicago, IL*	**AIAA-94-3754** **Linear Quadratic Regulator Problem with Control Inequality Constraints for Flexible Space Structures** J. Junkins and S. Lee, *Texas A&M Univ., College Station, TX*	**AIAA-94-3755** **Effect of Geometric Stiffness on Control of Robots with Flexible Arms** A. Banerjee, *Lockheed Palo Alto Research Lab, Palo Alto, CA;* M. Lemek and V. Do, *Sunnyvale, CA*	**AIAA-94-3756** **An Approach to Dynamics and Control of Flexible Systems** T. Nagata and H. Matsuo, *Univ. of Tokyo, Kanagawa, Japan;* V. Modi, *Vancouver, British Columbia, Canada*	**AIAA-94-3757** **An Active Vibration Suppression Method for the Flexible Spacecrafts and its Applications** D. Liu and G. Wu, *Harbin Inst. of Technology, Harbin, China*		

Wednesday Morning / August 3, 1994

Plenary Session **Future Flight Training Systems 8:00 am – 9:00 am** **Estrella Theatre**
Robert Barthelemy, Product Group Manager, Air Force Aeronautical Systems Center, Wright-Patterson AFB, OH

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 59-GNC-27 GN&C Components and Avionics Chaired by: D. BATES JR., C. S. Draper Laboratory, Cambridge, MA							**Room 7**
AIAA-94-3685 **Advances in High Performance Fiber Optic Gyroscope Systems** W. Taylor, *Honeywell Space Systems, Clearwater, FL*	AIAA-94-3686 **Inertial Navigation Performance of an IFOG Over Dynamic Environments** R. Patterson, E. Golder, A. Cordova, and D. Rozelle, *Litton Guidance and Control Systems, Woodland Hills, CA*	AIAA-94-3687 **Micromechanical Tuning Fork Gyro Test Results** M. Weinberg and A. Kourepenis, *C. S. Draper Lab, Cambridge, MA*	AIAA-94-3688 **An Electrostatic Gyroscope for Spacecraft Use** A. Cohen, *NASA Goddard, Greenbelt, MD*	AIAA-94-3689 **Engineering Test Satellite VII Optical Rendezvous Docking Sensor System** M. Mokuno, I. Kawano, H. Horiguchi, and K. Kibe, *NASDA, Tokyo, Japan*	AIAA-94-3690 **A Study of Optimal Design of Test Plan for Identifying the Model of Accelerometer on a Precision Centrifuge** G. Wu, F. Jiang, and D. Liu, *Harbin Inst. of Technology, Harbin, China*		
Session 60-GNC-28 General Aviation and Helicopter Applications Chaired by: G. SACHS, Technical University of Munich, Munich, Germany							**Room 8**
AIAA-94-3691 **Machine-Vision Based Pilot Aids for Night Landing and Takeoff** G. Chatterji, P. Menon, and B. Sridhar, *NASA Ames, Moffett Field, CA*	AIAA-94-3692 **Robust Rotorcraft Flight Control System Design Including Rotor Degrees of Freedom** P. Gorder and R. Hess, *Kansas State Univ., Manhattan, KS*	AIAA-94-3693 **Nonlinear Control of Rotorcraft Using Approximate Feedback Linearization and Online Neural Networks** J. Leitner, J. Prasad, and A. Calise, *Georgia Inst. of Technology, Atlanta, GA*	AIAA-94-3694 **Implicit and Explicit Model-Following Helicopter Flight Controllers** Z. Mirfakhraie and W. Garrard, *Univ. of Minnesota, Minneapolis, MN*	AIAA-94-3695 **Optimal Category-A Helicopter Takeoff from a Runway** Y. Zhao and R. Chen, *Univ. of Minnesota, Minneapolis, MN*			
Session 61-AFM-16 High Speed Aerodynamics Chaired by: J. SAHU, U.S. Army Research Lab, Aberdeen Proving Ground, MD							**Room 1**
AIAA-94-3519 **A Computational Study of the Aerodynamics of a Body Immersed in a Supersonic Wake** C. Ober and T. Kiehne, *Inst. for Advanced Technology, Austin, TX;* J. Lamb, *Univ. of Texas at Austin, Austin, TX*	AIAA-94-3520 **Side Force of Blunted Circular Cylinders at High Angles of Attack in Supersonic Flow** T. Yoshinaga, A. Tate, and H. Sekine, *National Aerospace Lab, Tokyo, Japan*	AIAA-94-3521 **Numerical Computations of Three Dimensional Jet Interaction Flow Fields** J. Sahu, *U.S. Army Research Lab, Aberdeen Proving Ground, MD*	AIAA-94-3522 **On the Sources of Aerodynamic Forces: Flow Around a Hemisphere-Cylinder** S-Y. Lei, *EBASCO-CTCI Corp., Taipei, Taiwan, ROC;* C-C. Chang, *National Taiwan Univ., Taipei, Taiwan, ROC*				
Session 62-AFM-17 Workshop on Flying Qualities Research and Pilot-Vehicle Interface Chaired by: R. MERCADANTE, Grumman, Bethpage, NY							**Room 2**
TBD							

Wednesday Morning / August 3, 1994

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 63-AFM-18 Trajectory Optimization Chaired by: T. SUMMERSET, The Aerospace Corporation, El Segundo, CA							**Room 3**
AIAA-94-3523 **A Generalized Technique for the Inverse Simulation of the Aircraft Motion Along Predetermined Trajectories** M. Abderlrahman and A. Al-Bahi, *King Abdulaziz Univ., Jeddah, Saudi Arabia*	**AIAA-94-3524** **Effect of Aeropropulsive Interactions and Design Sensitivities on Optimal Hypersonic Ascent Trajectories** D. Schmidt and T. Lovell, *Univ. of Maryland, College Park, MD*	**AIAA-94-3525** **Approximate Optimal Planar Guidance of an Aeroassisted Space Vehicle** D. Mishne, *Rafael, Haifa, Israel;* Y. Laufer, *Technion, Haifa, Israel*	**AIAA-94-3526** **Guidance and Control Strategies for Aeroassisted Orbital Transfer: Status Survey** D. Naidu, *Idaho State Univ., Pocatello, ID*	**AIAA-94-3527** **Explicit Guidance for Aeroassisted Orbital Plane Change** D-M. Ma, *Tamkang Univ., Taiwan, ROC*	**AIAA-94-3528** **Achieving the Maneuverability Domain Limits by Optimizing the Aircraft Trajectories** S. Manolescu, *Inst. of Fluid Mechanics & Flight Dynamics, Bucharest, Romania*		
Session 64-FST-8 Visual Systems Chaired by: J. BLEAK, Evans & Sutherland, Salt Lake City, UT							**Room 6**
AIAA-94-3437 **The Use of Low Cost Graphics Workstations as Simulator Image Generators** D. Paterson, *Marconi Simulation, Fife, Scotland, UK*	**AIAA-94-3438** **Toward More Realism in NVG Use in Training Simulators** P. Rapp, *Thomson-CSF, Osny, France*						
Session 65-FST-9 Standardization and Interoperability Initiatives (panel discussion—10:00 am) Chaired by: R. BURTON, Naval Air Warfare Center, Patuxent River, MD							**Ballroom E**
		Panelists: J. Shiflett, *USA, STRICOM, Orlando, FL* D. Bartlett, *USMC, DMSO, Washington DC* R. Hanley, *NAVAIRSYSCOM, Washington DC* M. Landry, *Lockheed, Ft. Worth, TX* W. Bezdek, *McDonnell Douglas Aerospace, St. Louis, MO*					
Session 66-ASD-11 Orbit Transfer Chaired by: F. HOOTS, General Research Corporation, Colorado Springs, CO							**Room 10**
AIAA-94-3758 **Analytically Exact Non-Keplerian Motion for Orbital Transfers** N. Markopulos, *Georgia Inst. of Technology, Atlanta, GA*	**AIAA-94-3759** **Transfer Mission Optimization of a Geosynchronous Spacecraft with a Solar Electric Arcjet Propulsion System** A. Schwer and U. Schoettle, *Univ. of Stuttgart, Stuttgart, Germany*	**AIAA-94-3760** **An Analytical Approach for Continuous-Thrust LEO-GEO Transfers** D. Spencer, *Phillips Lab, Kirtland AFB, NM;* R. Culp, *Boulder, CO*	**AIAA-94-3761** **Earth to Moon Transfer with a Limited Power Engine** M. Guelman, *Technion, Haifa, Israel*	**AIAA-94-3762** **Indirect Optimization Method for Impulsive Transfers** G. Colasurdo and D. Pastrone, *Politechnico di Torino, Torino, Italy*	**AIAA-94-3763** **Illustrative Orbital Applications of Electrodynamic Propulsion** C. Spenny, *Wright-Patterson AFB, OH*		

Wednesday Morning / August 3, 1994

9:00	9:30	10:00	10:30	11:00	11:30	12:00	12:30
Session 67-ASD-12 Mission Design and Analysis: Planetary Chaired by: R. HOLDAWAY, Rutherford Appleton Laboratory, Chjilton, UK							**Room 11**
AIAA-94-3764 **A Consistent Approach to Orbit Determination and Photogrammetry Utilizing Mars 94 Radio Tracking and HRSC/WAOSS Images** O. Montenbruck and E. Gill, *DLR, Wessling, Germany;* T. Ohlof, *Munich, Germany*	**AIAA-94-3765** **Mars Sample Return—A Low Cost, Direct and Minimum Risk Design** P. Wercinski, *NASA Ames, Moffett Field, CA*	**AIAA-94-3766** **Mars Free Return Trajectories** M. Patel, J. Longuski, and J. Sims, *Purdue Univ., West Lafayette, IN*	**AIAA-94-3767** **Analysis of Mapping Coverage Obtained from Spacecraft** A. Petropoulos, *Cambridge Technology Partners, Lansing, MI;* J. Longuski, *Purdue Univ., West Lafayette, IN;*	**AIAA-94-3768** **Characterization of Orbital Trajectories Between Earth and Mars Using Continuous Acceleration of 1/1000G** M. Deveny, *Carpenter Deveny Engineering, Corona, CA*	**AIAA-94-3769** **Analysis of V_∞ Leveraging for Interplanetary Missions** J. Sims and J. Longuski, *Purdue Univ., West Lafayette, IN*		
Session 68-ASD-13 Trajectory Optimization Chaired by: R. GLICKMAN, Ball Aerospace Systems, Boulder, CO							**Room 12**
AIAA-94-3770 **Study of Optimal Ascent Trajectories** S. Vadali and S-Y. Park, *Texas A&M Univ., College Station, TX;* H. Srywald, *NASA Langley, Hampton, VA*	**AIAA-94-3771** **Optimal Station Change Maneuvers for Geostationary Satellites Using Constant Low Thrust** N. Titus, *Phillips Lab, Kirtland AFB, NM*	**AIAA-94-3772** **A Search for Low Delta V Earth to Moon Trajectories** H. Pernicka and D. Scarberry, *San Jose State Univ., San Jose, CA;* S. Marsh, *Sunnyvale, CA;* T. Sweetser, *Pasadena, CA*	**AIAA-94-3773** **Interplanetary Trajectory Optimization Using a Genetic Algorithm** P. Gage, R. Braun, and I. Kroo, *Stanford Univ., Stanford, CA*	**AIAA-94-3774** **Number of Stations Trade-Off For Direct Ascent to GEO Launching** J-S. Chem, *CSIST, Lungtan, Taiwan, ROC;* Z-C. Hong, *Chungli, Taiwan;* C-Y. Chang, *National Central Univ., Chungli, Taiwan, ROC*	**AIAA-94-3775** **Optimal Low-Thrust Escape from the Solar System** G. Colasurdo and D. Pastrone, *Politechnico di Torino, Torino, Italy*		
Session 69-GNC-29 GN&C Education (panel discussion) Chaired by: J. VAGNERIS, University of Washington, Seattle, WA							**Ballroom F**
Panelists: Daniel Debra, Stanford University Steven Hall, Massachusetts Institute of Technology David Schmidt, University of Maryland Robert Skelton, Purdue University, Juris Vagners, University of Washington Jeanine Hunter, San Jose State University							
Session 70-GNC-30 GPS Issues (panel discussion) Chaired by: F. BAUER, NASA Goddard Space Center, Greenbelt, MD							**Ballroom G**
Panelists: Kurt Brock, Space Systems/Loral Bradford Parkinson, Stanford University Martin Pozesky, Federal Aviation Administration Penny Saunders, NASA Johnson George Schauer, Motoroa, Inc. Frank Bauer, NASA Goddard							

Flight Simulation Technologies Conference

TABLE OF CONTENTS

Paper No. **Title and Author** **Page No.**

FST 1

FST 2

N/A - Not available W/D - Withdrawn

FST 3

FST 4

N/A - Not available W/D - Withdrawn

N/A - Not available W/D - Withdrawn

N/A - Not available W/D - Withdrawn

AIAA-94-3400-CP

IMPROVEMENTS TO THE NAVAL AIR WARFARE CENTER AIRCRAFT DIVISION'S F/A-18 SUBSONIC AERODYNAMIC MODEL

Timothy R. Fitzgerald*
Strike Aircraft Test Directorate, Code SA103
Naval Air Warfare Center Aircraft Division
Patuxent River, Maryland 20670

John N. Ralston†
Bihrle Applied Research, Inc.
18 Research Drive
Hampton, Virginia 23666

Bruce L. Hildreth‡
Science Applications International Corporation, Office 827
100 Exploration Park, Suite 2005
Lexington Park, Maryland 20653

Abstract

An improved subsonic aerodynamic model has been developed for the F/A-18 simulation housed at the Manned Flight Simulator (MFS) facility located at the Naval Air Warfare Center Aircraft Division (NAWC AD). Advanced analysis and wind tunnel techniques were used to create this model which accurately duplicates aircraft motions inside the low-speed flight envelope, as well as beyond the fringes of that portion of the envelope including departures from controlled flight and spins. The development of this improved aerodynamic model will allow rapid and accurate support of Naval flight test requirements and fleet incidents.

Introduction

The increased capabilities of today's advanced tactical aircraft have become a double-edged sword for the United States Navy by increasing the overall costs associated with the procurement and service introduction of a weapon system. Initially, all the advanced technologies that are used when designing and building a new tactical aircraft increase development costs. While these advanced technologies greatly increase capabilities, thus reducing the number of units necessary to perform a mission, they also increase aircraft complexity. This results in expanded flight testing requirements and adds further to costs. Faced with this upwardly-spiraling trend in a continually shrinking defense budget, the U.S. Navy has increasingly turned to flight simulation to help defray flight test costs.

It is widely accepted that, for an aircraft simulation to be useful as a flight test tool, it must be reasonably accurate across the entire flight envelope. Lately, however, the realization has occurred that not only must the simulation faithfully reproduce the aircraft motion inside the flight envelope, but just beyond it as well (i.e., departures from controlled flight and spins). When discussing the high-speed end of the flight envelope, the problem is simplified by the fact that the edges of the envelope are governed, in general, by low angle-of-attack aerodynamics and structural and propulsion constraints, making it relatively easy to construct accurate simulation aerodynamic models using only traditional wind tunnel techniques. However, the aerodynamic capabilities of newer tactical aircraft, such as the F/A-18, have made modeling of the low-speed end of the envelope increasingly important, particularly at very high angles-of-attack and sideslip.

The F/A-18 aerodynamic model obtained by NAWC AD in 1983 from the airframe contractor adequately modeled the majority of the normal flight regime for the single-seat configuration. However, evaluation of simulation results versus flight test did reveal some areas of deficiency, particularly at high angles-of-attack and sideslip near the edge of the low-speed portion of the flight envelope. The model also did not contain sufficient aerodynamic data to simulate departures of the two-seat aircraft with centerline tank. An effort to introduce a more complete aerodynamic data base into the model and "unify" the two-part, up-and-away model was undertaken by NAWC AD, Bihrle Applied Research, Inc. (BAR), and Science Applications International Corporation, formerly Systems Control Technology, Inc. (SCT). The new aerodynamic data tables incorporate the results of parameter identification (PID) analysis as well as the aerodynamic increments associated with rotation about the velocity vector (rotary balance wind tunnel data). In addition, the effect of sideslip was expanded throughout the baseline model's functionality. The result is an aerodynamic data base continuous in angle-of-attack ($-90° \le \alpha \le 90°$), Mach number ($0 \le M \le 2$), and sideslip ($-30° \le \beta \le 30°$) with representative modeling of the fringes of the low-speed envelope for both the single-seat and two-seat aircraft.

It should be noted that the validation effort discussed here primarily involved the low-speed, higher angle-of-attack flight regimes of the F/A-18. Subsequent to this effort and currently underway, a more comprehensive evaluation seeks to validate the entire UP/AUTO flight envelope. This effort is being conducted jointly by NAWC AD, McDonnell Douglas Aerospace (MDA), and BAR in an attempt to address any other remaining simulation fidelity issues, and ultimately merge the best portions of the respective NAWC AD and MDA models into a single, global F/A-18 aerodynamic data base.

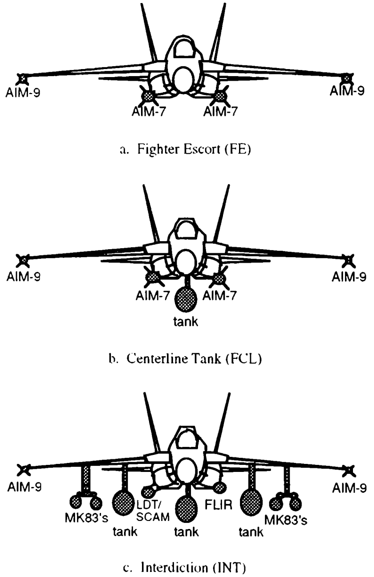
This paper shall discuss the processes involved in the creation of the NAWC AD F/A-18 unified aerodynamic model, and present validation results, recommendations, and lessons learned.

* Aerospace Engineer; Member AIAA
† Engineering Manager
‡ Site Manager; Member AIAA

History of the Model

A baseline simulation UP/AUTO (i.e., cruise) aerodynamic model was originally created as far back as 1974 by the McDonnell Douglas Corporation (MDC) using data supplied by Northrop from the YF-17. The first static wind tunnel testing on the F-18 configuration was conducted in 1976 by MDC using 6% and 16% scaled models at various wind tunnel facilities, with data for the dynamic derivatives collected using a forced oscillation technique. During subsequent data reduction by MDC, it was determined that the 16% model data was the most representative of the full-scale aircraft, and was thus favored for use in the simulation rigid body aerodynamic model. Aeroelastic flex/rigid ratios (based on theoretical and empirical analysis), drag data, and the high angle-of-attack data were subsequently added, such that by August 1977, MDC had a workable simulation aerodynamic model covering the majority of the UP/AUTO flight envelope for angles-of-attack from -4° to 90° in the fighter escort (FE) store loading[1].

In late 1983, the entire MDC F/A-18 simulation was acquired by NAWC AD, implemented into the MFS simulation architecture, and verified. Subsequent updates were received from MDC providing aerodynamic increments for the two-seat canopy, centerline tank (FCL) store loading, interdiction (INT) store loading, and leading edge extension fences, as well as to correct deficiencies in the original drag model. But by 1988, it became clear, through validation efforts and real-time use by NAWC AD, that the low-speed aerodynamic model contained inaccuracies. Figures 1a through 1c display the main store loadings used by the F/A-18 simulation, and are provided for reference.



a. Fighter Escort (FE)

b. Centerline Tank (FCL)

c. Interdiction (INT)

Figure 1 — Simulation Store Loadings

Additional Low-Speed Wind Tunnel Testing

Under contract from NAWC AD in 1988, BAR began static wind tunnel and rotary balance data collection and analysis on a 1/10-scale F/A-18 model in the National Aeronautics and Space Administration (NASA) Langley Research Center 20-foot spin tunnel. Data was collected for a total of seven different store loadings on the single-seat configuration, with the two-seat configuration initially tested in the FE loading only. However, it was later determined that additional wind tunnel testing was required on the two-seat configuration, both with and without the centerline tank (TFCL and TFE respectively), in order to more accurately reflect the incremental effects of these configuration changes on the baseline single-seat aerodynamic model used in the simulation. All combinations of canopies and store loadings were tested under static and rotational conditions both upright and inverted. A complete description of the initial BAR data collection and analysis effort may be found in References 2 through 4; a complete description of the follow-up BAR effort on the two-seat configuration may be found in Reference 5.

The rotary balance testing was primarily conducted to provide incremental effects due to wind axis rotation on the force and moment coefficients (see Reference 6 for complete description of test techniques and applications). This data has been successfully utilized in other simulations, using the techniques described in References 7 and 8, to accurately model aircraft spins and recoveries. Since this simulation improvement effort was to address this flight regime, these incremental effects were incorporated into the original MDC model. Initial validation of this hybrid aerodynamic model revealed poor correlation with the known F/A-18 spin characteristics, and as a result, an extensive review of all static data in the model, initially focusing on the flight regime encompassing departure and beyond, was undertaken. As mentioned earlier, additional testing and model development work was conducted as other deficiencies (e.g., two-seat departure modeling) became obvious.

Scope of the Parameter Identification Tests

Parameter identification of aerodynamic coefficients from flight test data was performed from 1985 through 1989. Using prototype aircraft F3, data was collected in the UP/AUTO configuration ($\alpha \le 55°$; $0.25 \le M \le 0.95$) for the single-seat F/A-18 with FE loading (wing tip AIM-9 missiles only). Two-seat F/A-18 data was also collected in the FE loading ($\alpha \le 40°$; $0.25 \le M \le 0.95$), using prototype aircraft TF1. In 1989, a test program was conducted involving the two-seat aircraft in four different store loadings: TFE (wing tip AIM-9 missiles only); TFCL (wing tip AIM-9 missiles only) plus two inboard pylons; TFCL (right wing tip AIM-9 missile only) plus two inboard pylons; and INT without the fuselage-mounted sensors/trackers. Once again using prototype aircraft TF1, data was collected during test maneuvers specifically aimed at PID analysis, while additional data was obtained from simultaneous non-PID flight test programs using aircraft TF30. This data covered a Mach range of 0.25 to 0.95 and altitudes of 25,000 to 40,000 feet, although the majority of the data was at Mach numbers less than 0.65. In all, a total of approximately 74 flights of two-seat data was made available for analysis. The majority of the data collected was at high altitudes, therefore the resulting analysis assumed an inflexible aerodynamic model.

It should be emphasized here that all the test data was not rigorously analyzed. Instead, maneuvers were chosen from each flight on the basis of best instrumentation quality and best excitation of the dynamic modes of the aircraft. Parameter identification software such as NAVIDNT/SCIDNT and Athena was used to perform the data analysis, and by 1986, reduction of the F3 flight test data was completed. The following year, the incremental aerodynamic effects for the two-seat canopy were estimated from the TF1 flight test data. References 9 and 10 contain complete descriptions of these PID efforts and their application to this model unification effort.

Aerodynamic Model Unification

Having collected the additional PID and wind tunnel test data, efforts were made to utilize this extensive new data pool to improve the original model. A number of specific tasks were required in order to accomplish this overall goal. First, a rigorous analysis of the model's deficiencies, both in structure and data, was conducted in order to define the required upgrade task. Second, a technique needed to be developed for analyzing and incorporating the diverse (and often non-compatible) data sets into the model in a fashion coincident with the model's multi-variable table look-up format. Finally, the unified data base would be re-validated and any remaining deficiencies identified and addressed.

Model data analysis was conducted by comparison with the other available data, including PID. This task required the development of model and data manipulation software that permitted the importation of simulation model, wind tunnel, and PID data bases, and through various matrix operations, reformation of each into compatible forms. This included such operations as center of gravity (c.g.) shifting, interpolation of additional breakpoint values, and the "delinearizing" of derivative terms, since, in most cases, functions were modeled non-linearly. Following this reformatting, the available data sets could be comparison-plotted and discrepancies noted. The PID data (where available) was then used as a guide to determine the most representative data set and the model data was adjusted to match. Where flight data was not available, further analysis, based on data source limitations and past experience was used to dictate ultimate model definition. Even though these limited cases did require "empirical" adjustments to the model to achieve the desired results, it was felt the model development based on wind tunnel data and its non-linear progression with angle-of-attack, sideslip, control surface deflection, etc. insured that the simulation data base reflected the most important data dependencies that would have been difficult or impossible to derive with any other method in the given flight regime (i.e., near- and post-departure). The following discussion describes some of the more pertinent data modifications made to the original modeling and their justifications.

Longitudinal

The initial PID analysis of F/A-18 flight data indicated that the simulation's longitudinal model was very good[9], which subsequently resulted in few model changes. These changes involved the revision of longitudinal dynamic derivatives and stabilator control power terms, and the overall structural changes that were made to the entire aerodynamic model. However, more recent analyses conducted independently by NAWC AD, BAR, and MDA has not shown the degree of correlation between the longitudinal aerodynamic model and the aircraft as the initial SCT analysis had. These model short-comings appear to be confined to the static portion of the aerodynamic model and are currently being addressed.

Lateral

Significant differences emerged in the analysis of the F/A-18's lateral characteristics, typical of that shown in Figure 2, which illustrates that in the stall/post-stall region, for the applicable sideslip range of the sloped data (±5°), the flight-extracted data differs substantially from the data contained in the original model. As can be seen, the static data obtained from the rotary balance test more closely approximates the flight-extracted terms. The lateral stability differences observed in these data are very similar to the conflicts noted in earlier F/A-18 test data taken from several tunnels at differing Reynolds numbers[11]. It was concluded at the time that the lateral characteristics as mechanized in the simulation gave the best representation of the full-scale airplane. However, the results of the flight-extracted data from several independent evaluations have indicated that the lateral stability for the F/A-18 should, in fact, be more stable than originally modeled. Consequently, these data were modified using a representative wind tunnel data set.

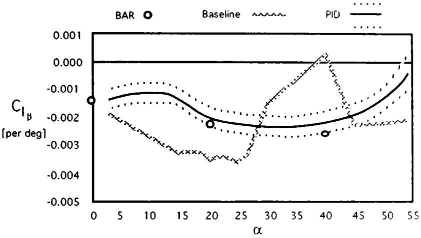


Figure 2 — Simulation Dihedral Effect Comparison

More crucial than the lateral characteristics for the basic F/A-18, however, was the mechanization of the two-seat F/A-18, particularly with the centerline tank. Original incremental effects of these two configuration changes were very limited in the simulation envelope, as well as limited in the resolution of the independent breakpoints. Substantial work, comparing available test data (i.e., References 12 and 13) as well as newly acquired data for this purpose, was conducted to extend and refine the incremental effects (e.g., Figure 3). The extension of the sideslip breakpoint set to reflect the non-linear effects of these configuration changes in the stall region was imperative because of the importance of these characteristics on the departure modeling of this configuration. In addition, for the lateral case as well as any coefficient with sideslip dependency, the sideslip effects were extended with wind tunnel data to ±30°. This was done because of the frequent high sideslip excursions this configuration experiences during departure, and the significance of these large sideslip effects throughout the aerodynamic model. More recent analysis has indicated that expansion of the sideslip dependency to ±45° will provide a further increase in model accuracy, as sideslips of this magnitude are not uncommon in some of the more violent departures.

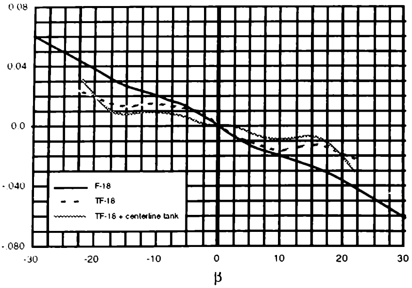


Figure 3 — Effects of Two-seat Canopy and Centerline Tank on Lateral Stability

Directional

The basic F/A-18 directional characteristics below 40° angle-of-attack exhibited generally good correlation with the flight-extracted data and consequently only minor changes were made. As in the lateral case, the two-seat canopy and centerline tank increments were significantly different than recent test data, and were subsequently revised to improve their definition at both low and high sideslip angles.

Analysis of the high angle-of-attack ($\alpha > 40°$) data revealed the lack of any modeled yaw asymmetries at zero sideslip. Several wind tunnel test data sets, including high Reynolds number test data (Figure 4), exhibited this behavior, a typical result of asymmetric forebody vortex shedding at these angles-of-attack. Further verification of the asymmetric yawing moment tendencies is found in free-spin model and flight test spin records in which a given airframe favored a particular spin direction. It was also felt this asymmetric behavior is one of the primary aerodynamic forcing functions in the low yaw rate spin. Consequently, asymmetry effects were included in the unified data base.

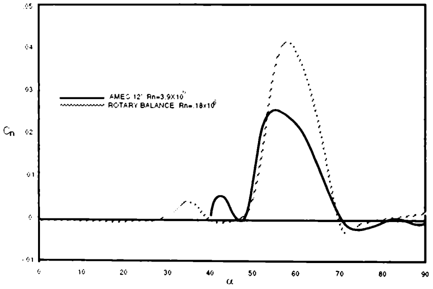


Figure 4 — Wind Tunnel Static Yawing Moment Comparison

Control Effects

Analysis of the control effects for the basic F/A-18 revealed a number of conflicts, particularly at high angles-of-attack. An example of this is shown in Figure 5, which compares test data with baseline simulation data in yaw for full aileron deflection. Although the baseline data is considerably more adverse (uncoordinating) in yaw, both data sets exhibit similar trends with angle-of-attack up through 60°, whereupon the baseline data breaks from adverse to proverse, and remains proverse through 90° angle-of-attack. This model data would suggest that pro-spin aileron deflection would be in the direction of the spin, opposite that of most military aircraft and inconsistent with F/A-18 flight test or free-spin tunnel data results. Therefore, the model data was modified to remove this high angle-of-attack effectiveness change.

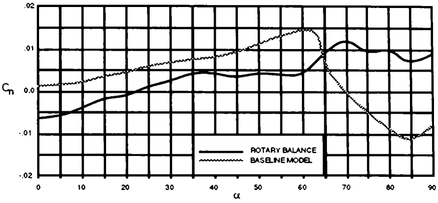


Figure 5 — Effect on Yawing Moment of a 50° Aileron Deflection

The modeled differential tail effectiveness in yaw was also examined versus other available test data. The incremental yawing moment that results from a differential tail deflection of -10° superimposed on either a 5° or -20° symmetric tail is presented in Figures 6a and 6b.

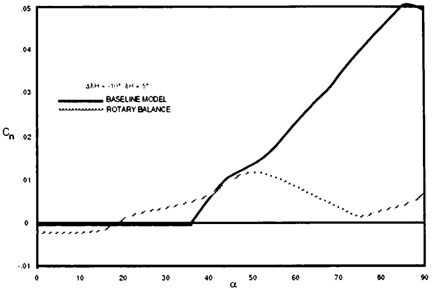


Figure 6a — Effect on Yawing Moment of a -10° Differential Tail Superimposed on a 5° Symmetric Tail

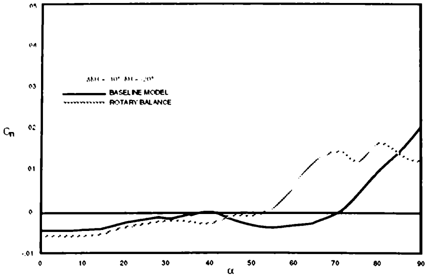


Figure 6b — Effect on Yawing Moment of a -10° Differential Tail Superimposed on a -20° Symmetric Tail

For the aft stick data, the yawing moments obtained are in good agreement below 50° angle-of-attack. Beyond 50°, the static data taken from the rotary balance tests becomes adverse, while the simulation data set remains proverse through 70° angle-of-attack, then rapidly becomes adverse. On the other hand, the yawing moment produced by the differential tail with full forward stick becomes increasingly more adverse above 35° angle-of-attack, ultimately reaching extremely high levels of adverse yaw. These characteristics, in conjunction with the aileron yaw authority characteristics previously mentioned, result in an simulation aerodynamic model with little or no propelling yawing moment for a combination of aft stick and pro-spin roll controls (i.e., stick against the spin), and very large amounts of propelling yawing moment for forward stick. The spin behavior observed in the free-spin tunnel[14] and flight test, as well as the spin modes predicted using rotary balance data[15] do not reflect these control characteristics.

At low angles-of-attack and for full rudder deflection, correlation between the simulation data and other data sources was very good. However, analysis of the two-seat F/A-18 low angle-of-attack departure data revealed the simulation's rudder control effectiveness was low for the maximum rudder deflections allowed by the flight control system (approximately 11°) at the departure condition. Test data taken with additional rudder deflections revealed rudder effectiveness was more dependent on deflection angle than was modeled and so this effect was incorporated into the simulation data.

With the exception of small sideslip effects on rudder deflection at high angles-of-attack, no sideslip effect was modeled in the baseline aerodynamic data for any control surface. Past analyses have shown the important influence of sideslip on control effectiveness, particularly at the angles-of-attack and sideslips experienced during departure (e.g., Figure 7, showing the effect of sideslip on rudder power). The complete non-linear modeling of these effects was included for all control deflections.

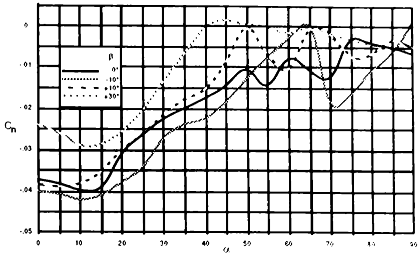


Figure 7 — Effect of Sideslip on Yawing Moment due to a 30° Rudder Deflection

Dynamic Derivatives

Comparison of the modeled dynamic derivatives with flight-extracted values revealed a number of conflicts. As a result, additional dynamic data were derived for the upright and inverted single-seat F/A-18 using a modified strip method, which uses the rotational slopes of the complete airplane, as well as airplane component data (both static and rotational) to formulate the derivatives. The derivative is calculated as pure rate terms (i.e., C_{l_p}) rather than a rate plus beta-dot term (i.e., $C_{l_p} + C_{l_{\dot\beta}} \sin \alpha$) that is typically the output of forced oscillation testing. The availability of this data set was useful in the further analysis and correlation of the dynamic effects, as illustrated in Figure 8, where the roll due to roll rate term (C_{l_p}) is compared for the available data sets.

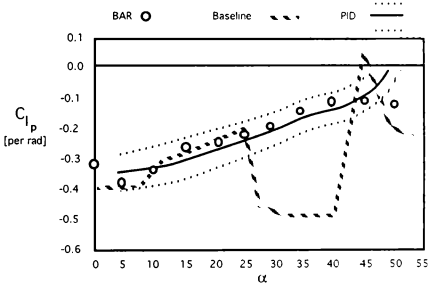


Figure 8 — Simulation Roll Damping Comparison

As shown by Figure 8, the flight-extracted data is considerably less damped than the baseline model in the stall region, but agrees well with the values calculated from the rotary balance tests. The sharp increases in the roll damping at stall exhibited by the original data set may have contributed in the selection of the original lateral characteristics described earlier, with the increased roll damping offsetting the reduced lateral stability evidenced by the baseline model at these angles-of-attack. As noted earlier, however, independent PID efforts have derived roll damping terms that are less damped than originally modeled, and other analysis has shown that this sharp increase in damping at stall may be a result of the inclusion of the $\dot{\beta}$ effect in the total roll damping term[16]. As a result, the unified model made use of the calculated data where dictated by the PID results. Differences in the other derivative terms were similarly rationalized and corrected during the model unification process.

Model Structure

Because the original data base consisted of two static data regions describing angle-of-attack envelopes of approximately -4° to 40° and 40° to 90°, a ramping function was required to smoothly transition between these data regions. This requirement arose because of data discrepancies at the table end points, as well as dissimilarity in the actual breakpoints and table geometries at the break. Additionally, an inverted static low-speed data base[4], as well as the rotational increments were to be combined to these two tables. Thus, following the revision of the actual upright data as described earlier, it was felt that this segmentation of the data base was no longer satisfactory, and consequently, the merging of all discrete data bases into a continuous, breakpoint-compatible form was conducted using the model manipulation tools. Many of the modifications made to the upright model itself had required reformatting of the -4° to 90° angle-of-attack region. Consequently, the addition of the inverted data resulted in a continuous simulation data base from -90° to 90° angle-of-attack, ±30° sideslip, and Mach numbers from 0.2 to 2.0.

Validation of the Unified Aerodynamic Model

Due mainly to NAWC AD project requirements for simulation support, much of the validation effort to date has been centered around the fidelity of the simulation in a fully developed spin, and the departure fidelity of the simulation in the TFCL configuration. As previously mentioned, a more comprehensive evaluation of the fidelity of the simulation's aerodynamic model, encompassing the entire UP/AUTO flight envelope, is currently underway.

The NAWC AD F/A-18 simulation contains a built-in validation utility, known as SCOPE (Simulation Checking using an Optimal Prediction Evaluation), that allows it to be over-driven with time history data collected from another simulation or from flight test. SCOPE can either over-drive the entire simulation with pilot commands (six degree-of-freedom) or individual models (e.g., the aerodynamics) with the appropriate parameters (zero degree-of-freedom). The upshot of this ability is to duplicate the inputs and conditions of the test maneuver exactly, thereby allowing one-for-one comparisons of the resulting time histories in order to determine model fidelity. For the purposes of this validation, both modes of SCOPE were used: for the six degree-of-freedom comparisons, pilot stick and rudder commands were fed into the simulation at the appropriate flight condition, and the resulting rates and attitudes compared with those collected from flight test; for the zero degree-of-freedom comparisons, measured flight conditions and control surface deflections were input into the aerodynamic model, and the resulting aerodynamic moment coefficients compared with those computed from the flight test data. In addition to the SCOPE comparisons, the simulation was evaluated with a pilot-in-the-loop in order to determine if the simulation exhibited the correct dynamic trends displayed by the aircraft during flight test. The techniques used for the pilot-in-the-loop evaluations were identical to those used in the respective flight test application: for TFCL departures, the simulation was flown to one of the eight documented departure points used for this evaluation and the departure-inducing controls applied; for spins, following a 1-g stall, pro-spin controls combined with the application of asymmetric thrust (low mode spin) or selection of the flight control system's manual spin recovery mode (intermediate and high mode spins) were used. Recovery techniques included both hands-off (i.e., neutral controls) and anti-spin. Additionally, maneuvers designed to explore the simulation's high angle-of-attack and inverted flying qualities as well as its ability to resist departures at conditions near a known departure point were flown. Evaluations of the simulation's departure resistance in the TFE, FCL, and FE configurations were also conducted, as were spins with several symmetric and asymmetric store loadings. Simulation time histories were then compared to those collected in flight test and comparisons of critical parameters such as angle-of-attack, sideslip angle, steady-state yaw rate, roll angle, and in the case of spins, turns to recovery, were made.

Discussion of Results

Simulation High Angle-of-Attack Fidelity

At or near 1-g conditions and for forward c.g.'s, the F/A-18's high angle-of-attack handling qualities are excellent. Even up to angles-of-attack of 55° to 60°, the aircraft is generally well-behaved and controllable. However, as the c.g. progresses aft, the aircraft is prone to a high angle-of-attack hang-up. This hang-up may also be initiated or aggravated by the presence of wing rock[17]. The F/A-18's wing rock is more accurately described as a reduction in directional stability characterized by an indistinct "nose-wander" above stall. This results in sideslip excursions which cause low-to-moderate roll oscillations.

In general, both the baseline and unified aerodynamic models exhibit both the benign high angle-of-attack handling qualities and wing rock tendencies of the F/A-18. One specific area of improvement over the baseline model, however, is the ability of the unified model to duplicate the aircraft's high angle-of-attack hang-up. Pilots did occasionally comment, however, that in the TFCL loading, the unified model seemed "loose" directionally at elevated angles-of-attack ($25° \leq \alpha \leq 35°$), and that the asymmetric rolling and yawing moments built into the unified model may be too strong when compared to the aircraft.

Simulation Departure Fidelity

The two-seat F/A-18 exhibits three regions of weak directional stability, especially with a centerline tank: low-speed, low angle-of-attack; low-speed, high angle-of-attack; and high subsonic Mach, low angle-of-attack[17]. The single-seat aircraft, even with centerline tank, has much greater directional stability in these regions, but will on rare occasions, depart. Once the F/A-18 departs, the incipient motions can range from relatively benign to very violent, and the subsequent post-departure gyrations are nearly unpredictable. Indeed, flight test has shown that very small differences in the departure entry conditions result in seemingly random post-stall motions that can vary radically from an almost immediate recovery to a protracted coupled pitch trim (i.e., "falling leaf") or low yaw rate spin. For the purposes of this evaluation, only the low-speed TFCL departures were investigated.

Figures 9a and 9b present comparisons of the baseline and unified aerodynamic models' lateral-directional coefficient errors that result from a zero degree-of-freedom over-drive of a TFCL departure into a "falling leaf" (see Figure 10 for an actual flight motion comparison).

In this case, an error of 0.0 would indicate a perfect match when compared to the coefficients extracted from the F/A-18D flight test aircraft (BuNo 163452). As shown in the plots, substantial improvement in the error propagation was achieved with the unified model, particularly at the early stages of the departure. This is shown more graphically by the six degree-of-freedom time history comparison presented in Figure 10. As seen in this figure, the baseline model exhibits only a benign sideslip build-up with no departure, whereas the unified model matches the initial departure motion very well. Even the post-departure motions are highly representative, with accurate modeling of both the amplitude and frequency of the rate excursion. Although the unified model recovers one cycle earlier than the aircraft in this case and exhibits a poor angle-of-attack match, this is likely the result of some of the pitch inaccuracies mentioned previously, in particular, the lack of sideslip effects on pitching moment.

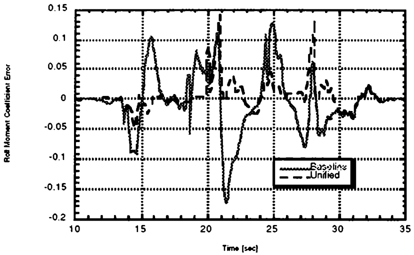


Figure 9a — SCOPE Zero Degree-of-Freedom Roll Moment Error Comparison (F/A-18D TFCL "Falling Leaf")

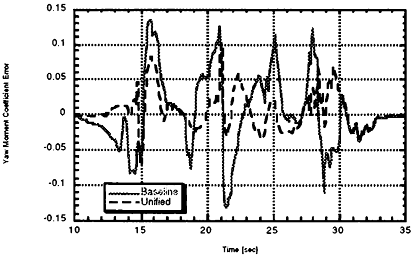


Figure 9b — SCOPE Zero Degree-of-Freedom Yaw Moment Error Comparison (F/A-18D TFCL "Falling Leaf")

During pilot-in-the-loop evaluations, the unified aerodynamic model was able to satisfactorily demonstrate the aircraft's departure entry characteristics and sensitivity to entry conditions, as well as the unpredictable post-departure motions experienced in the TFCL loading. These motions varied from immediate recoveries to more protracted post-stall motions. The simulation would periodically emulate the aircraft's tendencies to roll inverted, then upright again during post-stall oscillations or re-depart after having apparently recovered. During particularly violent departures, entry into a "falling leaf" has also been experienced. In general, the departure and post-departure motions were deemed very representative for the various F/A-18 loadings examined, a particularly important characteristic for the use of a simulation of this type in the support of flight test.

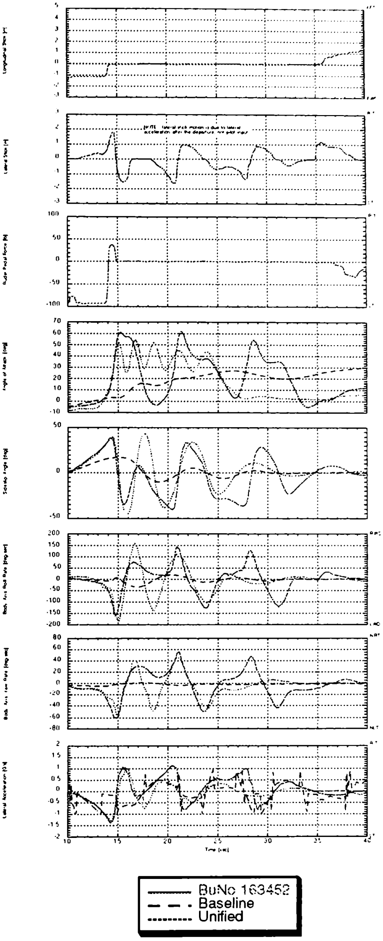


Figure 10 — SCOPE Six Degree-of-Freedom Time History Match (F/A-18D TFCL Departure and "Falling Leaf")

Simulation Spin Fidelity

Once in a fully developed spin, the F/A-18 typically exhibits pitch, roll, and yaw oscillations that vary in intensity between its three spin modes. The low yaw rate mode is characterized by mild pitch, roll, and yaw oscillations, although the mode can be quite smooth as well. The intermediate yaw rate spin is usually quite oscillatory in all three axes, particularly in roll where it is not uncommon for roll oscillations to build to the point where the aircraft will execute a full 360° body axis roll while continuing to spin in the established direction. Finally, the high yaw rate spin is generally smooth and flat, but moderate oscillations in all three axes can occur. Detailed descriptions of the F/A-18 spin modes are contained in References 17 and 18.

Tables I and II compare steady-state results of individual low, intermediate, and high yaw rate spins performed using the simulation with generalized flight test results.

SPIN MODE	yaw rate [deg/sec]	α [deg]
Low	25	42-54
Intermediate	35-70	30-90
High	125+	72-90

Table I — F/A-18 Simulation Spin Results (FE loading)

SPIN MODE	yaw rate [deg/sec]	α [deg]
Low	0-40	30-60
Intermediate	20-80	55-90
High	90-140	70-95

Table II — F/A-18 Generalized Flight Test Spin Results (FE loading)[17, 18]

Figure 11 presents a six degree-of-freedom time history comparison between prototype aircraft F6 (BuNo 160780) and the unified aerodynamic model for an intermediate yaw rate spin.

Overall, the unified aerodynamic model was able to duplicate the aircraft's three spin modes very well. Characteristic oscillations for each of these spins were accurately modeled, and variations in rate and oscillations with particular longitudinal and lateral control combinations was also observed (i.e., an intermediate yaw rate spin could be driven into a high yaw rate spin if attention to lateral control application was not maintained). The addition of a two-seat canopy had no discernible effect on spin characteristics, which is consistent with flight test results. Simulated spin motions with both symmetric and asymmetric store loadings were also deemed representative of the F/A-18, as the simulation exhibited the aircraft's tendencies to recover with neutral controls in the FE loading, but remain auto-rotative with an asymmetric loading of 10,000 ft-lb or greater. Recovery characteristics and timing with anti-spin controls were also considered very representative of the full-scale aircraft, as recoveries were accomplished within 1-1/2 turns for low and intermediate yaw rate spins, and 2-1/2 turns from a high yaw rate spin[18]. Finally, asymmetric store loadings did not significantly increase the number of turns for the simulation to recover (approximately 1 additional turn to recover), which is also consistent with the aircraft. One minor discrepancy noted during the pilot-in-the-loop evaluations was that the simulation entered the low yaw mode spin more readily than the aircraft. However, this is most likely due to optimistic modeling of the engine thrust rather than an error in the aerodynamic model, since asymmetric thrust is used to drive the auto-rotation, and when a reduced throttle setting was used, the simulation response was more accurate.

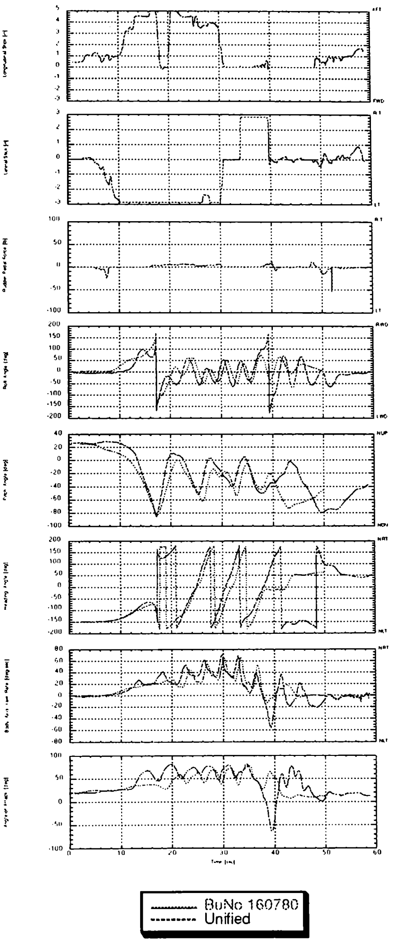


Figure 11 — Six Degree-of-Freedom Time History Comparison of Intermediate Yaw Rate Spin (F-18A FE loading)

Summary and Conclusions

The fidelity of the NAWC AD simulation has been enhanced through the development of the unified aerodynamic data base, as demonstrated through its ability to faithfully duplicate F/A-18 out-of-control motions such as departures and spins. Similar increases in simulation fidelity have been observed within the flight envelope for $M \leq 0.6$.

While the unification process has increased the size of the data base considerably, current computer capabilities no longer place severe limits on data base size. As a result, the access to and visualization of coefficient propagation is improved, and subsequent model development work will be greatly eased.

This task has demonstrated the feasibility of accurately modeling not only the in-control phase of flight, but also the out-of-control motions. Where previous attempts to model these motions have centered around substantial empirical "adjustment" to the model using flight data (or even pilot comments) as a guide, this effort has demonstrated the integration of PID and wind tunnel data to develop an accurate, large angle, six degree-of-freedom aerodynamic model. An analysis process, focusing on the applicability of the data to the selected flight regime and motions using PID results as a guide, was developed to intelligently apply the wind tunnel data resources to the formulation of the final unified aerodynamic model. In this fashion, the requirements for both the large amplitude statics and dynamic damping characteristics in the modeling of these motions were established. In addition, the technique for implementing the dynamic effects described in Reference 7 was successfully used in the modeling of these motions.

Recommendations

Continue to eliminate remaining known deficiencies as part of the data base integration effort between NAWC AD and MDA. This should include the implementation of any further corrections necessary to the parts of the F/A-18's flight envelope not covered in the original unification effort: longitudinal UP/AUTO ($M < 0.6$); UP/AUTO transonic and supersonic ($M > 0.6$); take-off (half flaps); and powered approach (full flaps).

Investigate further the pilot perceptions of directional "looseness" and overly-strong lateral-directional asymmetries with the unified model at elevated angles-of-attack.

Continue to evaluate new methods of integrating the small perturbation (i.e., forced oscillation) and large amplitude (i.e., rotary balance) damping characteristics into the simulation environment. An ongoing analysis sponsored by NAWC AD has shown the methodology described by Kalviste[8] appears particularly promising.

Acknowledgments

The authors would like to express their gratitude to employees of Bihrle Applied Research, Inc. and members of the Flight Systems Department, Strike Aircraft Test Directorate for their contributions to this simulation update effort.

References

1. Hodnefield, K. M.: "History of F-18 Aero. Data Updates"; informal MDC memo, c. 1977.

2. Ralston, John N.: "Analysis of F-18A/B Low-Speed, High Angle-of-Attack Aerodynamic Data Base"; BAR 88-7, September 1988.

3. O'Connor, Cornelius: "NATC F-18A/B Aerodynamic Math Model Modifications Incorporated during the Phase I Model Unification Effort"; BAR 89-5, March 1989.

4. O'Connor, Cornelius: "NATC F-18A/B Aerodynamic Math Model Modifications Incorporated during the Phase II Model Unification Effort"; BAR 90-13, September 1990.

5. Ralston, John N.; Avent, John L.: "Evaluation and Modification of F/A-18B Aerodynamic Data Base for Improved Departure Modelling"; BAR 92-2, March 1992.

6. Ralston, John N.; Barnhart, Billy P.: "Application of Rotary Balance Data in the Analysis of High Angle-of-Attack Characteristics"; NASA Conference Publication 3149, Part 3, 1992.

7. Bihrle, William; Barnhart, Billy: "Spin Prediction Techniques"; Journal of Aircraft, Vol. 20, No. 2, February 1983.

8. Kalviste, J.: "Use of Rotary Balance and Forced Oscillation Test Data in a Six Degree-of-Freedom Simulation"; AIAA 82-1364, August 1982.

9. Hess, Robert A.: "Subsonic F/A-18A and F/A-18B (TF-18A) Aerodynamics Identified from Flight Test Data"; SCT 4522-220-1, July 1987.

10. Hess, Robert A.: "Development and Evaluation of a High-Fidelity F-18 Aerodynamic Model in a Unified Format"; SCT 6007-070-1, October 1989.

11. Erickson, Gary E.: "Water Tunnel Flow Visualization and Wind Tunnel Analysis of the F-18"; NASA CR 165859, May 1982.

12. Mason, D. H.: "Investigation of the RECCE Nose on the TF/A-18 High Angle-of-Attack Stability in the David Taylor Naval Ship Research and Development Center 7-ft. Transonic Wind Tunnel"; MDC B 2053, July 1990.

13. Grafton, Sue B.: "Static Tests to determine Aerodynamic Effects of a Twin Canopy Configuration of the F-18"; unpublished NASA data report, May 1987.

14. Scher, Stanly H.; White, William L.: "Spin Tunnel Investigation of a 1/30-Scale Model of the McDonnell Douglas F/A-18 Airplane"; NASA TMX SX-81809, August 1980.

15. Hultberg, Randy: "Low Speed Rotary Aerodynamics of F-18 Configuration for 0° to 90° Angles-of-Attack"; NASA CR 3608, August 1984.

16. Ogburn, Marilyn E.; Nguyen, Luat T.; Hoffler, Keith D.: "Modeling of Large Amplitude High Angle-of-Attack Maneuvers"; AIAA CP-88-4357, August 1988.

17. "NATOPS Flight Manual: Navy Model F/A-18A/B/C/D"; A1-F18AC-NFM-000, 15 January 1991; change 5 — 15 August 1993.

18. McNamara, William G. et al: "Limited Flight Systems Navy Technical Evaluation of the F/TF/A-18A Airplane, Final Report, of 17 Mar 1983"; NAVAIRTESTCEN SA-126R-82.

MATCHING A FLIGHT TRAINING DEVICE WITH A REAL AIRCRAFT USING A GENTIC ALGORITHM

Reinhard Braunstingl
Technical University of Graz, A-8010 Graz /Austria
E-Mail: braun@fmechds01.tu-graz.ac.at

Abstract

The aim of providing realistic, type-specific flight simulation reaches financial limits rather than technical ones. It is a well known fact that these limits are rather low for small pilot schools in General Aviation. Therefore, the method here presented tries to meet the needs of General Aviation by matching the simulated stationary flight performance of flight training devices with a specific aircraft at minimum costs. The method takes its starting point from a given generic aerodynamic model for fixed-wing aircraft. The basis for determining the initially open parameters of the model is provided by flight measurings which can be carried out very easily and with a low budget as only standard equipment instruments of the aircraft are involved in the process. Using a genetic algorithm which searches for values giving optimum convergence between the simulated and measured flight performances the model is then adapted to a particular aircraft. To reach quick convergence using only few individuals the optimisation process has to be done in steps by looking separately at flight conditions with flaps and landing gear up and down respectively.

1 Introduction

Flight simulators are an ideal complement to practical pilot training. Many manoeuvers can be trained more easily, at lower costs and taking less risks than in an aircraft. Who for example will ever set an aircraft's engine on fire just to practice how to extinguish it? Of course, the value of flight simulator training strongly depends, among other things, on how well the simulator imitates the system and the flight behaviour of the aircraft in question. Given adequate technical effort, both aspects can be marvellously realized. The resulting costs however, very soon reach the economic limit, especially in General Aviation. For example, nobody can imagine a Level-D flight simulator for a Cessna 182; not for technical reasons but for economic ones. Therefore, pilot schools for aircraft of categories A (single engine aircraft up to 2000kg) to C (multi-engined aircraft up to 5700kg) do not use flight simulators but so-called flight training devices, which have to meet by far fewer requirements. (Although those training devices are often called simulators as well, in this paper the terminology conforming with the regulations of the Federal Aviation Administration, FAA is used and a line drawn between flight simulators levels A-D and flight training devices levels 1-7).[1,2] Flight training devices up to level 5 for example, do not require an aircraft-specific aerodynamic model, i.e. those devices can simulate a fictitious aircraft which is typical of the corresponding class of aircraft. For a trainee passing from flight training device to aircraft and viceversa always involves some accomodation process, as the performance of the training device never exactly squares that of the aircraft used in training.

Flight training devices use more or less complicated models for simulating aircraft movement. In order to reflect the performance of a particular aircraft, aerodynamic data of that aircraft is necessary which has to be the more detailed the more exact the simulation has to be. For the majority of aircraft in General Aviation such exact data, however, is not available. Of course, it could be investigated but the costs involved would be irresponsibly high. This is the point where the method described here comes in. An aerodynamic model and a thrust model [3] are both designed in a way that by choosing a few, basic parameters, they are able to describe the performance of any conventional fixed-wing aircraft for all flight conditions essential in pilot training. For model verification only those performance data of the aircraft are used which can easily and therefore cost-efficiently be determined, and which nevertheless are sufficient to adapt the simulation model well to the aircraft. The degree of accuracy of simulation which can be achieved by this method is so high that pilots recognize their training aircraft in the flight training device without difficulty.

Of course, limiting measuring flights to production aircraft involves limitations with respect to verification and validation of the model. Parameters for lateral motion, for example, cannot be registered quantitatively and accelerated flight conditions also escape validation. The training routine in pilot schools, however, has shown that this does not represent a serious limitation, as in the actual cockpit the pilot has no possibility - i.e. no instruments - to quantify lateral motion either. However, purely qualitative conformity of the model with reality is absolutely possible for these flight conditions and has proved sufficient. In areas where a pilot can define flight performance exactly and where he expects it from the flight training device (stationary climb and descent with varying configurations, cruising at varying engine performances) the model can be adapted well and easily - as will be shown below - to performance data imposed by reality.

The method here presented was developed with regard to the needs of General Aviation and has been put into practice in flight training devices built by the Austrian simulator production company *bt-austria*. All those flight training devices for aircraft of different categories, reaching from a PIPER PA28 to a CESSNA CITATION II, have been approved by the Austrian and German Federal Offices of Civil Aviation (Österreichisches Bundesamt für Zivilluftfahrt, BAZ and Luftfahrtbundesamt, LBA respectively) and are used in pilot training.

2 Stationary flight in the vertical plane

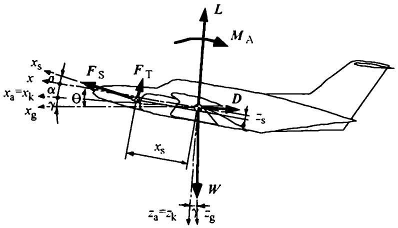


Figure 1: Stationary flight in the vertical plane at zero wind

Figure 1 shows an aircraft at stationary flight in the vertical plane at zero wind. All forces and moments must be in a state of equilibrium which can be expressed as follows:

$$\begin{aligned} -D+F_S\cos(\alpha+\sigma)-F_T\sin(\alpha+\sigma)-W\sin\gamma &= 0 \\ -L-F_S\sin(\alpha+\sigma)-F_T\cos(\alpha+\sigma)+W\cos\gamma &= 0 \\ M^A+(F_S\sin\sigma+F_T\cos\sigma)x_S+(F_S\cos\sigma-F_T\sin\sigma)z_S = M &= 0\,. \end{aligned} \tag{1}$$

Using the non-dimensional forces and moments C_L, C_D, C_S, C_T, and C_M^A and together with the reference quantities wing area S, propeller disc area S_P, wing chord l_μ, and dynamic pressure $q=(\rho/2)V^2$ we get

$$\begin{aligned} -C_D+[C_S\cos(\alpha+\sigma)-C_T\sin(\alpha+\sigma)]\frac{S_P}{S}-\frac{W}{qS}\sin\gamma &= 0 \\ -C_L-[C_S\sin(\alpha+\sigma)+C_T\cos(\alpha+\sigma)]\frac{S_P}{S}+\frac{W}{qS}\cos\gamma &= 0 \\ C_M^A+(C_S\sin\sigma+C_T\cos\sigma)\frac{S_Px_S}{Sl_\mu}\ + & \\ +(C_S\cos\sigma-C_T\sin\sigma)\frac{S_Pz_S}{Sl_\mu} = C_M &= 0\,. \end{aligned} \tag{2}$$

If we insert mathematical models for the aerodynamic coefficients here, which exactly represent a given aircraft the right-hand side of equation (2) reads zero for any stationary flight. Unfortunately only nature herself knows the exact form of the models here required. Therefore, we can only use mere approximations which result in the best possible convergence of the simulation with both flight measurements taken before and the data in the flight manual. It is the target of the optimisation process described below to reach this convergence. To get this result we use equation (2) to calculate numerically three variables, namely angle of pitch, vertical speed and elevator deflection for various flight conditions and compare them with the measured data. Then a genetic algorithm searches for those model parameters which result in optimum accordance between simulated and measured data with all flight conditions taken into consideration.

3 Flight measurings

Part of the performance data needed as reference for the validation process can be found in the flight manual of the aircraft. Normally flight manuals do not contain performance data for all possible stationary flight conditions. In most cases data for the angle of pitch for a particular flight condition are not available, so that additional measuring flights are required. However, these measuring flights are carried out only with production aircraft, in which no reequipments have to be undertaken, as only the normal flight instruments are being used as measuring instruments. Therefore, these measuring flights are as cost-efficient as can be. Nevertheless it is possible to determine with sufficient accuracy all data necessary for adapting the model.The demands imposed on the pilot are not very high either. He only has to be able to adhere - not so much exactly but absolutely stationarily - to a given flight speed with varying engine performances and configurations.

During such a measuring flight stationary sideslip free flight conditions are reached in succession while the positions of flaps and landing gear are varied and then the display of the following instruments recorded on video camera: clock, airspeed indicator, artificial horizon, vertical speed indicator, manifold pressure gauge, engine speed counter, and outside air temperature gauge. The screen-filling shot of each instrument ensures excellent readability and troublefree evaluation of the shots. The respective positions of flaps and landing gear can be recorded on the sound track of the video tape. The average amount of time needed for the transition from one stationary flight condition to another, i.e. the amount of time needed to reduce or increase speed and to record the data - for each measuring point - takes about two minutes according to experience. For the adaptation of the aerodynamic model of a PIPER PA-28RT-201T TURBO ARROW IV, described below, 175 measuring points were gathered this way.

4 Optimisation

The optimisation task consists in finding those values for the open parameters of the aerodynamic model for which the flight performance of the flight training device and that of the given aircraft match best. For this purpose we have to define a function of those parameters showing the degree of convergence in a quantitative way. For the method at hand a function is chosen guided by the way pilots judge a flight training device: normally pilots fly the flight training device at various altitudes, at various speeds, varying engine power and configuration and then compare vertical speed and angle of pitch with the corresponding values of the aircraft known from experience. As a result three variables of state, namely non-dimensional vertical speed w_g^*, angle of pitch Θ, and elevator deflection η are calculated from the three equations (2) for each measuring point and the difference between the measured and the simulated values determined. The non-dimensional vertical speed $w_g^*=w_g/V_R$ is the vertical speed relative to the reference speed of the aircraft $V_R=[2mg/(\rho S)]^{1/2}$. In this way the following coefficient for the total deviation D_{tot} of the flight training device is defined

$$D_{tot} = \frac{1}{n}\sum_{i=1}^{n} \left| g_w(\Delta w_{g_i}^*)^2 + g_\Theta(\Delta\Theta_i)^2 + g_\eta(\Delta\eta_i)^2 \right| \quad , \tag{3}$$

where n is the number of measuring points and g_w. g_Θ and g_η are weights. Considering that pilots perceive a deviation Δw_g^* of 0.03 and Δ_Θ of 0.05 radians as equally serious it is not too difficult to estimate the relation of the weights g_Θ/g_w. An arbitrary choice of $g_w=1$ results in $g_\Theta=0.36$. As has been mentioned before. the data published in the flight handbook do not contain any data concerning angle of pitch. For these measuring points therefore g_Θ was set to 0. The third weight g_η is always set to 0, which means that the difference between calculated and actual elevation deflection is not taken into consideration. When taking the measurings, the value for elevator deflection is gone without as in a real aircraft the pilot has no possibility to determine the elevator's position either and therefore a quantitative comparison is impossible. The purely qualitative correctness of the elevator deflection, however, is already safeguarded by the model, which is absolutely sufficient. Testing the model in flight training devices confirmed this: No pilot voiced doubts concerning the correctness of elevator deflection.

It is the task of the optimisation process to find the minimum total deviation, i.e. the global minimum of equation (3) if possible. As the genetic algorithm used searches for the maximum of a function, the inverse value of D_{tot} is used as a so-called fitness function which measures the fitness of a set of parameters U_i of the aerodynamic model and thus the quality of the flight training device

$$f(U_1,U_2,U_3,\ldots,U_{16}) = \frac{1}{D_{tot}} \quad . \tag{4}$$

Choosing unfavourable model parameters could result in equation (2) not being fulfilled for any value of w_g^*, Θ or η lying within reasonable limits. If this holds true for only one measuring point, f is set to 0, as the aerodynamic model with this set of parameters does not allow stationary flight for at least one measuring point and this set of parameters is definitely useless. It is to be expected that the fitness function is zero for big parts of the parameter space, which makes deterministic optimisation procedures abort immediately. A genetic algorithm, however, can relatively quickly find areas of interest in the parameter space. Given several local optimum values it can also jump from one to the next - if the size of population is big enough - whereas deterministic optimisation procedures have to be re-started, hoping to hit the catchment area of a higher optimum value. For this reason a genetic algorithm was chosen for this task.

4.1 The Genetic Algorithm

Genetic algorithms are the result of attempts to imitate the most important principles of biological evolution, which obviously is a most powerful optimisation process. A multitude of abstract, artificial individuals are looked at simultaneously. Each new generation of these individuals is created by combining anew elements of the best indiviuals. Now and then mutations happen. In contrast to the so-called evolutionary strategies, genetic algorithms do not work with numerical values but with strings. [4] They imitate natural evolution, which does not change the individuals themselves either, but from generation to generation simply combines the genes of the best. The biological equivalent of these strings would be the genetic material, i.e the genotype. We get the so-called phenotype only when the string is decoded and in our case yields a set of desired parameters of the aerodynamic model, which is submitted to the stress of selection by its environment. This selective effect of the environment is represented by the fitness function, which assigns a fitness value to each set of parameters. Whereas biological evolution produces indiviuals which are adapted as well as possible to their environment, the genetic algorithm leads to sets of parameters with the highest possible fitness value, in other words to sets of parameters which match the flight training device as closely as possible with the aircraft.

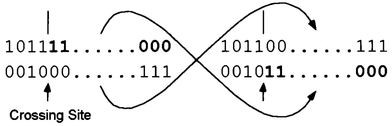


Figure 2: Crossover - an operator of the genetic algorithm. The two strings represent the genetic material of two individuals. They are cut open at a randomly chosen position and the two pieces swapped thus producing two new individuals.

The *Simple Genetic Algorithm* [5] used here imitates the most important mechanisms of biological evolution by means of the following three operations: reproduction, crossover, and mutation. At first the parental generation is reproduced by copying every string with a likelihood proportional to its fitness value. Therefore "better" individuals have a higher chance of reproduction than not so good ones. Then the strings are grouped in pairs, cut open at any position and the pieces swapped (figure 2). This process called crossover is carried out with constant likelihood. After this every sign of a string is mutated, i.e. arbitrarily modified, with a much lesser likelihood. The mutation process is not regarded as the driving force of optimisation. Its role is limited to bringing back into it those pieces of information which were lost by unfavourable crossover. For the rate of mutation you can and should, therefore, choose a value which is the lower the bigger the number of individuals.

An additional operation, linear fitness scaling, is used to reduce the danger of converging on a local optimum. As each individual gets copied into the next generation at a likelihood rate which is proportionate to its fitness, an individual with excellent fitness would place too many copies of itself in the next generation. If this happens at an early stage of the evolution, all the individuals would immediately concentrate on one area of the parameter space in which there may be just a low, local optimum thus preventing the indivuals from leaving this area again. That is why a linear, scaled fitness f' of all individuals is calculated in a way that the fitness of the best indiviual f_{mx} is a C- fold value of the average fitness f_{av}, which remains unchanged during scaling ($f'_{av}=f'$).

$$f' = \frac{f_{av}}{f_{mx}-f_{av}}[f(C-1)+f_{mx}-Cf_{av}] \qquad (5)$$

A set of parameters was coded using "multiparameter coding" with a binary code.

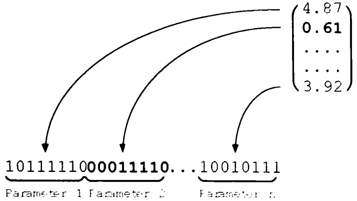


Figure 3: The parameter vector of the aerodynamic model is coded into a string by changing every numerical value into a binarily coded integer. The resulting string represents the socalled genetic information of an indiviual.

To do this we have to choose a range $U_{imin} \le U_i < U_{imax}$ for each parameter U_i and every value within this interval is then mapped onto an unsigned integer I ranging from 0 to 2^k.

$$I = Int\left(\frac{U_i-U_{imin}}{U_{imax}-U_{imin}}2^k\right) . \qquad (6)$$

Coded binarily and placed one after the other these integers yield a sting. In our case k=15 was chosen, resulting in strings of 240 characters each for the 16 parameters of the aerodynamic model. A number of such strings makes up a population.

4.2 Progess and results of optimisation

How this optimisation process works is illustrated with the help of a widely used aircraft of General Aviation, a PIPER PA-28RT-201T TURBO ARROW IV. In the process, two parameters of the genetic algorithm are kept constant: the crossover probability is set to 0.9 and the factor of fitness scaling $C=2$, i.e. the fitness of the best individual of each generation is always twice as high as the average fitness.

At first all 16 parameters of the aerodynamic model are to be determined simultaneously. Using a population of 3000 individuals and a correspondingly low rate of mutation of 0.001 the genetic algorithm converges on a solution with a fitness value of approx. 5900 (Figure 4) after 250 generations. Applying this set of parameters the mean deviation of the aerodynamic model reads 65 fpm for vertical speed and 0.6 degrees for angle of pitch. This means that the stationary flight performance of the flight training device is already very close to that of the aircraft. However, this solution represents only a local optimum and apart from that it needed 750,000 fitness function calls.

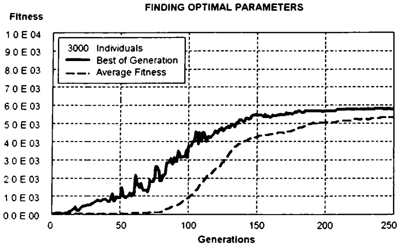


Figure 4: If the genetic algorithm is to find all parameters of the aerodynamic model simultaneously, the population converges only on a local optimum, even if the number of individuals is very high.

It does not make any sense to carry the calculation beyond 250 generations as the population has already become too homogeneous, that is to say, it consists of a great number of strings which closely resemble each other. In principle, genetic algorithms are able to leave a local optimum again, but this can become very unlikely if the population has clustered in a very small area of the parameter space. The increase in average fitness is used here as a criterion to determine whenever this is the case. If average fitness reaches a similarly high value as the fitness of the optimum individual, there must already be a great number of individuals in immediate proximity to the best one. Therefore, the chance of again leaving this area of the parameter space decreases, as the strings also resemble each other more and more and crossover no longer can help to make accessible a new region. The genetic algorithm is caught at a local optimum.

An even better solution can be found a lot faster if the search for the optimum model parameters is carried out in two separate subspaces. For this purpose the flight conditions for clean configuration with flaps and gear up are looked at separately from the flight conditions with flaps and gear down.

In the first case, five model parameters are inefficient, namely those which take into account the change of lift and drag caused by flaps and gear thus limiting the search area to a subspace of 11 dimensions. Now a population of 200 individuals is already sufficient to find an even higher optimum with far fewer fitness function calls. The rate of mutation, however, must be increased to 0.01 as crossover eliminates essential genetic information at an early stage with smaller populations.

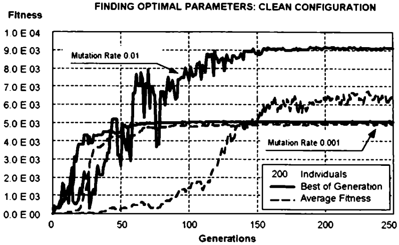


Figure 5: The optimisation in a subspace quickly leads to a high optimum already with a population of 200 individuals. If the rate of mutation is too low, the population converges even faster but on a lower local optimum.

Figure 5 clearly illustrates this. If the mutation rate is too low, high fitness values are reached at very quickly, but the population rapidly becomes homogeneous. The (unscaled) average fitness is almost as high as that of the best individual of the generation.

Afterwards the values for 11 model parameters thus arrived at are kept unchanged and only those parameters considered which are relevant for the changes of lift and drag due to flaps and gear. This results in an optimisation task in an even smaller parameter space of only 5 dimensions for which again a population of 200 individuals is chosen (figure 6). Once more the mutation rate of 0.001 proves too low while a value of 0.01 achieves a good result.

Increasing the mutation rate further than 0.01 did not lead to better results. It only showed that fitness was subjected to even greater fluctuation which could become so big that even areas with high fitness are left altogether. The population is too small to cling to an optimum if big disruptions caused by mutation occur.

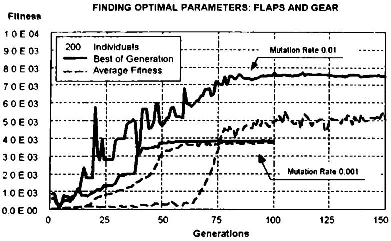


Figure 6: Only the parameters relevant for flaps and gear are once more looked for in a subspace. Again, with to small a rate of mutation the danger of premature convergence arises. The set of parameters thus determined finally represents a mean deviation of 52 fpm with regard to vertical speed and 0.5 degrees with regard to angle of pitch between flight training device and aircraft.

The set of parameters thus determined shows a fitness of 8.100 with regard to all flight conditions, which is even higher than the one shown in figure 4, and was found with the help of only 80,000 fitness function calls. The difference in vertical speed between the flight training device and the aircraft, therefore, is only 52 fpm on average, and the difference regarding angle of pitch is 0.5 degrees, which is already within the area of obtainable measuring accuracy for the equipment described. A flight training device that has been equipped with a model that well adapted to a particular aircraft, gives the pilot the chance to realistically train flight procedures as they find the same performance both in the flight training device and the aircraft.

References

[1] Federal Aviation Administration, "Airplane Flight Training Device Qualification", in: *Advisory Circular*, Nr. 120-45A, U.S. Department of Transportation, Oct. 4,1990.

[2] Federal Aviation Administration, "Airplane Simulator Qualification", in: *Advisory Circular*, Nr. 120-40B, U.S. Department of Transportation, (Proposed).

[3] Braunstingl, R., "Ein aerodynamisches Modell für die Längsbewegung von Flugübungsberäten", in: *Zeitschrift für Flugwissenschaften und Weltraumforschung*, Dec. 1992, p. 363-369.

[4] Schöneburg, Heinzmann, Feddersen, *Genetische Algorithmen und Evolutionsstrategien*, Addison Wesley, 1989.

[5] Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Reading, Massachusetts, 1989.

AIAA-94-3403-CP

OPTIMAL SIMULATOR AERODYNAMIC MODEL DEFINITION USING A GENETIC ALGORITHM

Douglas D. Hensley
Boeing Commercial Airplane Group
Renton, Washington

Abstract

This paper describes the use of a Genetic Algorithm in defining an optimal simulator aerodynamic math model. Using a Genetic Algorithm, a simulator model was generated for an airplane derivative and a new airplane simulator update scenario. The simulator models generated conceptually agreed with expected results and were at data levels comparable to the engineer defined results. The use of a Genetic Algorithm could significantly reduce the time required for a simulator model update.

Nomenclature

$C_{L_{SIM}}$	Simulator total lift coefficient, stability axes
$C_{L_{FT}}$	Flight test total lift coefficient, stability axes
ΔC_L	Lift coefficient difference between required and computed value, stability axes
K	Number of points defining the flight test data target surface

Introduction

The objective of the aerodynamic simulator model update process is to converge the simulator aerodynamic math model to the actual airplane aerodynamic characteristics within specified tolerances. The key to a successful simulator update are the methods used for extracting aerodynamic simulator data from flight test time history data.[1] Within the Boeing Commercial Airplane Group (BCAG), the methods of data extraction utilize powerful computer-based tools, but require considerable interpretation and judgement of an engineering team who have extensive knowledge of the aerodynamic effects on aircraft performance, stability, and control[1]. This process is a highly iterative task requiring a significant commitment of engineering and computer resources. By introducing computer automation into the intermediate iterations of the update process, the time and resources required for a simulator update could be significantly reduced.

Current Simulator Update Process

The current simulator update process relies extensively on an engineer's interpretation and judgement to iteratively update the simulator aerodynamic math model. The simulator math model is implemented as a build-up of aerodynamic force components. For example, the total simulator aerodynamic lift, in airplane stability axes, can be expressed as[1]:

$$\begin{aligned} C_{L_{SIM}} = {} & C_{L_{BASIC}} + C_{L_{AEROELASTIC}} \\ & + C_{L_{STABILIZER}} + C_{L_{ELEVATOR}} \\ & + C_{L_{DYNAMIC}} + \text{SMALLER TERMS} \end{aligned} \tag{1}$$

During each iteration, the total error between the simulator model and the flight test data is determined:

$$\Delta C_L = C_{L_{FT}} - C_{L_{SIM}} \tag{2}$$

Where the flight test total lift coefficient of equation (2) is calculated from the forces and moments required to satisfy intertial equilibrium for the kinematic state and mass characteristics determined from flight test measurements.

If the total error exceeds specified limits, the engineer's interpretation and judgement are used to assign the error and make the required changes to specific members and regions of the force component models of equation (1). The effect of these changes on the total error is then evaluated. Typically modifications are made in a sequential progression through the build-up components. The use of multiple engineers updating various terms of the simulator model simultaneously and the cross-coupling effect inherent in the simulator model necessitates coordination of modifications to other simulator build-up terms. It is the highly iterative nature of this process that results in the need for a significant commitment of engineering and computer resources.

Automated Simulator Update Process

Background

Minimizing the error between two surfaces using an iterative convergence technique is a problem ideally suited for computer automation. Generally an automated surface convergence technique would require a definition of the surfaces to converge and a cost-function to optimize. The simulator aerodynamic math model represented in equation (1) defines the surface to be converged. The flight test data, or an appropriate model representing the flight test data, provide the target surface. Minimizing the least square error of equation (2) provides an appropriate cost-function:

$$\text{Min}\left[\sum_{i=1}^{K}(\text{FT}_{\text{SURFACE}} - \text{SIMULATOR}_{\text{SURFACE}})^2\right] \quad (3)$$

Where:

K = Number of points defining the flight test data target surface

Implementation

Genetic Algorithm Overview

The surface convergence technique used in this study was a Genetic Algorithm. The Genetic Algorithm has been established as a robust and efficient optimization technique across a broad spectrum of problems. The Genetic Algorithm derives its name from its basis upon the theory of natural selection. A Genetic Algorithm generates populations of potential solutions of the convergence problem. The suitability of each potential solution is evaluated using a cost-function, e.g. equation (3). The population members best satisfying the cost-function are operated upon by three genetic operators: selection, crossover, and mutation to generate successively improving generations of solutions. Reference 2 is an excellent resource for understanding Genetic Algorithms.

Genetic Algorithm Implementation

The chromosome structure used to encode the population of potential solutions is depicted in Figure 1. Each gene within a chromosome represents a specific force component model from the simulator aerodynamic math model's build-up terms, such as basic lift and aeroelastic effect on lift. Each gene is composed of alleles; an allele represents the dependent value at each breakpoint in the force component model. These dependent values were encoded into a binary representation with 15-bit resolution (8-bit in Figure 1 example). Thus each potential solution chromosome is a concatenated binary string encoding of the dependent values of the simulator force component models to be modified.

Genetic Algorithm Enhancements

The Genetic Algorithm used in this study had some variation from the simple Genetic Algorithm of Reference 2. These enhancements improved the convergence characteristics of the algorithm.

The initial population was not generated randomly over the entire range of bit resolution. It was assumed that a predicted simulator model was available and the final model would exist near the region of the predicted model. Thus the initial population was randomly chosen within a local region defined for each force component model.

The mutation and cross-over probabilities were allowed to vary based upon the rate-of-improvement exhibited in successive generations of potential solutions. When the rate-of-improvement was low, indicating a homogeneous population, the cross-over probability would be decreased and the mutation probability increased. The increased mutation probability injected diversity into the population. Conversely, when the rate-of-improvement was high, indicating a heterogeneous population, the mutation probability would be decreased and the cross-over probability increased. The increased cross-over probability increased the beneficial effect of inter-mingling within the diverse population.

When the population was homogeneous and the mutation probability was high, the population would be re-initialized. This population re-initialization immediately restored diversity into the population while benefitting from the results of previous generations. This re-initialization was accomplished similar to the method of generating the initial population, however the population was now chosen in the region of the current best potential solution rather than in the region of the predicted simulator model.

Results

Overview

The Genetic Algorithm described was used to generate modified force component models for a simulator aerodynamic math model. These Genetic Algorithm defined models were generated for two scenarios: an airplane derivative and a new airplane simulator update.

The first scenario was intended to represent the small force component model changes typical of a simulator update for an airplane derivative program. In this scenario, the Genetic Algorithm population was initialized using the simulator model of the baseline airplane. A comparison of the initial model, the engineer defined final model and the Genetic Algorithm generated model for basic lift and aeroelastic effect on lift are shown in Figures 2 and 3, respectively. The Genetic Algorithm results very closely agree with the engineer defined results.

The second scenario was intended to represent the large force component model changes typical of a simulator update for a new airplane program. The basic lift and aeroelastic effect on lift component models used previously were scaled to 85 percent to represent the sometimes large differences between predictions based upon wind-tunnel data and the final simulator model. Figure 4 shows a comparison between the modified initial model, the engineer defined final model and the Genetic Algorithm generated model for basic lift.

In the region where flight test data were available, the Genetic Algorithm generated model was converging toward the engineer defined model. As the angle of attack increases, the Genetic Algorithm results appear to converge less strongly. This characteristic has two causes. First, the scaling factor applied to the initial model introduced increasingly pronounced differences with increasing values of basic lift coefficient. Second, the Genetic Algorithm's search was too strongly constrained to the local region of the initial model. Thus as the differences between the initial model and the expected final model increased, the search constraints significantly decreased the likelihood of the Genetic Algorithm to find a solution far from the initial model. The sudden transition of the Genetic Algorithm model back to the initial scaled model at the highest angles of attack indicate regions where flight test data did not exist.

Discussion of Results

The efficacy of an automated surface convergence technique for modifying the simulator force component models can be evaluated on 3-levels:

- Conceptually
- Data Level Comparisons
- Proof-of-Match Requirements

On the conceptual level of evaluation, the question to be answered is:

- Do the force component models produced conform with expected characteristics? In other words, does the basic lift model look like a C_L-vs-α curve?

A brief study of the results presented in Figures 2 - 4 clearly show the Genetic Algorithm generated force component models conform with expected curve shape characteristics. In fact, the aeroelastic effect models (Figure 3) generated by the Genetic Algorithm demonstrate a greater level of uniformity than the engineer defined models.

The evaluation of the data level requires two questions to be answered:

- How do the force component models produced compare with the engineer produced models?
- How consistent are the force component models produced beginning at one start point compared with those produced when begun from a different start point?

The force component models (Figures 2 - 4) generated by the Genetic Algorithm very closely agree with the engineer defined final component models data levels. Even when the initial error in the force component models was very large (Figure 4), the Genetic Algorithm produced model definitions were converging toward the engineer defined models. Figure 5 shows a comparison of the Figure 2 and 4 force component models generated by the Genetic Algorithm when the starting points were significantly different. This comparison shows that in the regions where flight test data exist the Genetic Algorithm will produce consistent results even when the starting points differ significantly.

The Proof-of-Match is the highest level of evaluating the automated simulator surface convergence method. At this level the question to be answered is:

- How well do the force component models satisfy the requirement for a final simulator model which satisfies the simulation qualification requirements for all types of maneuvers throughout the flight envelope?

The force component models generated by the Genetic Algorithm in this study were not evaluated at the Proof-of-Match level of evaluation.

Conclusions

The process of updating the aerodynamic simulator math model is a highly iterative task requiring a significant commitment of engineering and computer resources. This iterative convergence process is amenable to computer automation. A Genetic Algorithm produced a simulator model very comparable to engineer defined results. Although an automated surface convergence technique can significantly reduce the time required for a simulator model update, an engineer's judgement will still be necessary to ensure a final simulator model which satisfies the requirements for all types of maneuvers throughout the flight envelope.

Acknowledgements

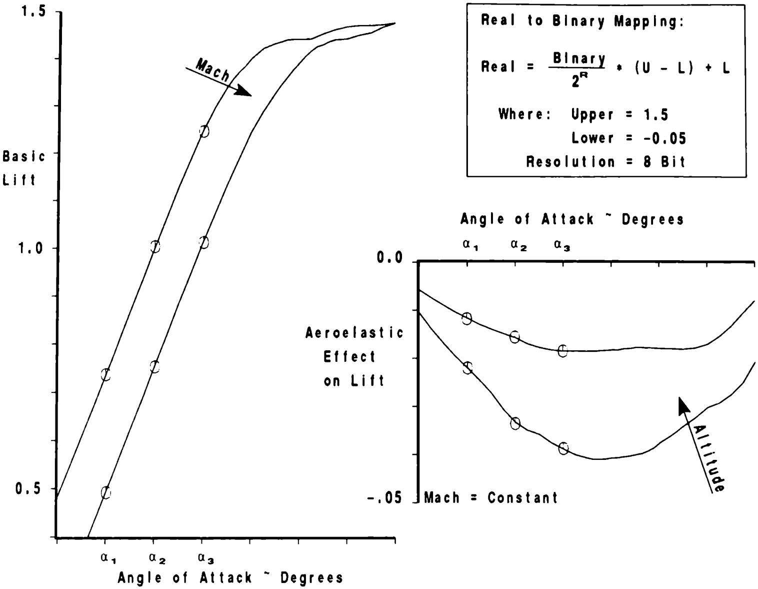
Very special thanks to Terry Schulze of the Boeing Commercial Airplane Group Aerodynamics Engineering for his enthusiastic support which made this work possible.

Special thanks also to Brian Bocksch and Brett Churchill for their support and encouragement.

References

1. Neville K.W. and Stephens A.T., Flight Update of Aerodynamic Math Model. In AIAA Flight Simulation Technologies Conference (Monterey, California 1993).

5. "Genetic Algorithms in Search, Optimization and Machine Learning", D. E. Goldberg, 1989, Addison-Wesley Publishing Company.



Real to Binary Mapping:

$$\text{Real} = \frac{\text{Binary}}{2^R} * (U - L) + L$$

Where: Upper = 1.5
Lower = -0.05
Resolution = 8 Bit

CHROMOSOME: Lift

GENE: Basic Lift			Aeroelastic Effect on Lift		
ALLELE:					
M_1	Decimal	Binary	H_1	Decimal	Binary
α_1	0.735	10000010	α_1	-0.022	00000101
α_2	1.0	10101101	α_2	-0.0334	00000011
α_3	1.25	11010111	α_3	-0.0385	00000010
M_2			H_2		
α_1	0.49	01011001	α_1	-0.0115	00000110
α_2	0.75	10000100	α_2	-0.016	00000110
α_3	1.05	10110110	α_3	-0.0185	00000101

Concatenated Chromosome Encoding:

Basic Lift: 100000101010110111010111010110011000010010110110 +
Aeroelastic: 000001010000001100000010000001100000011000000101

FIGURE 1, Genetic Algorithm Chromosome Structure

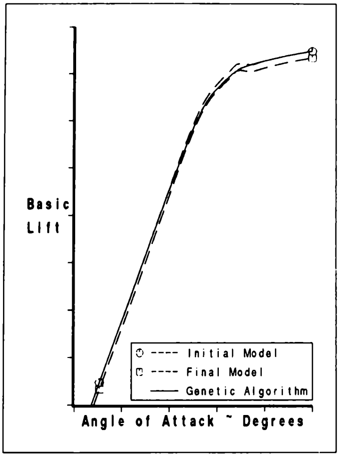


FIGURE 2. Basic Lift - Genetic Algorithm Results Comparison

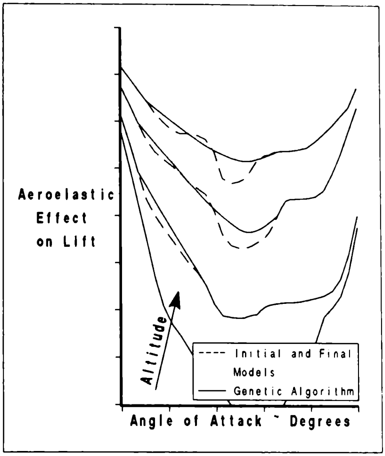


FIGURE 3. Aeroelastic Lift - Genetic Algorithm Results Comparison

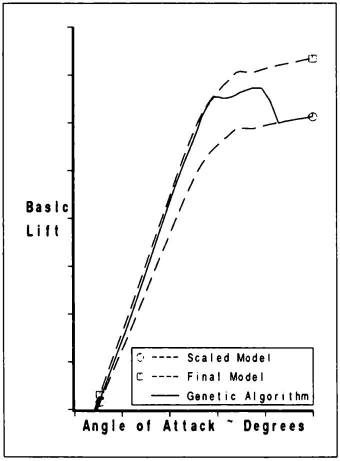


FIGURE 4. Scaled Basic Lift - Genetic Algorithm Results Comparison

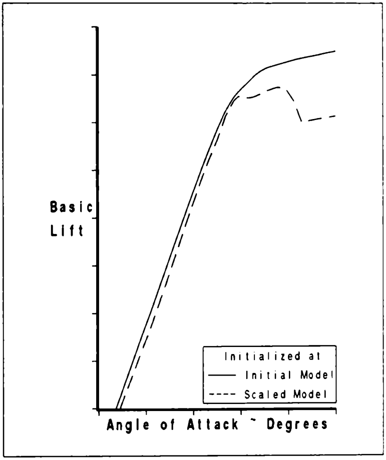


FIGURE 5. Basic Lift - Genetic Algorithm Consistency

AIAA-94-3404-CP

ASPECTS OF C-160 SIMULATOR MODEL DETERMINATION AND VALIDATION ON AND CLOSE TO THE GROUND

D. Fischenberg,* W. Mönnich,** B. Krag, * R.V. Jategaonkar *

Institut für Flugmechanik
Deutsche Forschungsanstalt für Luft- und Raumfahrt (DLR)
Lilienthalplatz 7, D-38108 Braunschweig
Germany

Abstract

Modern flight simulators demand accurate mathematical models because of increasingly stringent qualification standards. Maneuvers with ground contact such as takeoff, landing, and taxiing are essential parts of simulator model validation with flight test data. During the validation, discrepancies often occur between simulated and measured data, whose analysis requires extended care to separate the aerodynamic, landing gear, and thrust effects. In this article the model updates for validation according to the FAA Level D standard on and close to the ground are discussed. Four examples are presented: single engine takeoff modeling is improved considerably taking into account the asymmetric thrust influence on aerodynamics. For landing validation, it is found to be essential to account for wind gradients. It is demonstrated that the aircraft derotation after touch down can be simulated properly only by accounting for the tire acceleration of the landing gear. For taxi validation, precise skid angle modeling of the undercarriage tires is found to be necessary.

Introduction

The C-160 "Transall", although being in service by the French and German Air Force since more than 20 years, will continue to be the main transport aircraft in the next decade. For pilot training, the German military procurement organization decided to procure a modern flight simulator, which will replace the old one at Wunstorf airbase.

Under contract, DLR Institute of Flight Mechanics developed the aerodynamic data base and mathematical model for this simulator, which is being built by the Simulator Division of Thomson-CSF according to the FAA Level D standard[1]. Although a lot of wind tunnel measurements are available for this 'old' aircraft, it was decided to carry out an extensive flight test program in order to derive a suitable mathematical model from flight test data. System identification methods were used to estimate this completely new aerodynamic derivative model, as this is nowadays - besides the update of the wind tunnel data base with flight data[2] - a common technique for determining a high quality simulator data base[3-5]. The data base not only incorporates the aerodynamic data valid within the operational flight envelope, but also a ground handling model with undercarriage kinematics and tire characteristics, and an engine and propeller model. The flight test program and the development of the purely airborne math model is described in ref. 6.

The mathematical simulator model had to be validated with flight test data according to the FAA defined validation tests, which were complemented with several C-160 specific maneuvers (e.g. load drop, short field takeoff and landing). A main part of the validation is the aircraft behaviour on and close to the ground, namely takeoff, landing, and taxiing. In this region, the model validation within the stringent FAA simulator qualification tolerances is a difficult and sensitive task[7,8].

This paper pays attention to the simulation and validation problems that arise on and close to the ground by combining the ground handling model, the engine model, and the aerodynamic model including ground effect. An insufficient simulation quality requires a careful analysis of each model in order to decide upon the 'right' model to be updated. Having decided upon the right model, the database is augmented with additive effects to simulate aircraft behaviour on and close to the ground within the validation tolerances. Some effects on simulation quality are discussed and a few C-160 validation plots of typical FAA requirements on and close to the ground are presented.

*Scientist
**Scientist, Affiliate Member AIAA

Aircraft and Undercarriage Design

The C-160 "Transall" (Fig. 1) is a twin turboprop powered aircraft with a maximum takeoff weight of 51000 *kg*. Its maximum cruising speed is 277 *kt* at an altitude of 16000 *ft*, its wing span is 40 *m*. It is equipped with the conventional aerodynamic control surfaces, namely elevator, aileron, rudder, landing flaps, air brakes, and spoilers. The two engines are equipped with four-bladed propellers both rotating in the same direction and thus influencing the aerodynamics in an asymmetric way. The engine control logic commands constant speed only between the power lever positions Flight Idle and Minimal Cruise, from this condition upwards to takeoff power the propeller speed is linearly increased by controlling the propeller pitch. That means there is a strong nonlinear function between the pilot's power lever inputs and the thrust response.

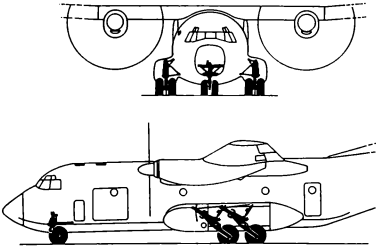


Fig. 1. C-160 "Transall"

The retractable landing gear consists of a two-wheeled, steerable nose gear and four two-wheeled main gears. The main gear struts are mounted with a considerable inclination angle to the fuselage reference line, and the oleo dampers are compressed by a rotating motion of the lower struts. Each of the main gears has an anti skid braking system, right and left brakes can be operated independently.

Airborne Aerodynamic Math Model

Applying a nonlinear maximum likelihood method, an aerodynamic model valid over the entire operational envelope was estimated from flight data[6,9]. The analysis started with point identifications, where aerodynamic derivatives were identified for small excursions around a trim point. This procedure was repeated for each evaluation point of the flight envelope and each of the five landing flap positions. Analysing these data, an extended derivative model was constituted taking into account Mach number, angle of attack, angle of sideslip, and thrust dependencies as well as cross coupling effects. In an iterative process, a multi-point evaluation considered up to 75 flight maneuvers for each landing flap position simultaneously. The model had to be changed several times in order to obtain an acceptable fit between measured and computed time histories. Finally, a derivative model was formulated that consists of about 70 parameters for each flap position.

Having determined this basic aerodynamic model, special effects like speed brake and landing gear influences, being important for takeoff and landing, were identified[10].

Aerodynamic Changes Due to Ground Effect

The ground effect has three main effects: a reduced downwash angle at the tail, an increase in the wing-body lift slope, and an increase in the tail lift slope[11,12]. These effects influence mainly the longitudinal motion, but the lateral-directional characteristics are also altered. In order to identify the C-160 ground effect, several flights were conducted at low altitudes (200, 100, 40, 20 *ft* above ground level) including maneuvers exciting the longitudinal and lateral motion. Unfortunately during these flight tests atmospheric turbulence was encountered, and as the aircraft dynamics could only be excited by small amplitude doublets, it was not possible to separate out conclusively the ground effect using these flight test data. At the end, it was decided to derive the ground effect modeling from the collected landing and takeoff recordings, although there are no defined control inputs for system identification.

For C-160, it was found that the ground effect can be modeled adequately as a linear function of altitude h. If the aircraft is rolling on the ground, the wing is about 4 m above the ground, which is nearly equal to the mean aerodynamic chord of the wing ($\bar{c} = 4.18\,m$). That means those altitudes, where the ground effect changes in a strong nonlinear way ($h < \bar{c}$)[12], will not be reached by the C-160 top-wing configuration.

The modeled incremental coefficients due to ground effect for the longitudinal aerodynamics are lift ($\Delta C_{L_{GE}}$), drag ($\Delta C_{D_{GE}}$), and pitching moment ($\Delta C_{m_{GE}}$) as a function of center of gravity (CG) altitude h and the maximal ground effect altitude h_{GE}.

$$\begin{aligned}
\Delta C_{L_{GE}} &= \frac{\partial C_L}{\partial h}(h - h_{GE}), \qquad h < h_{GE} \\
\Delta C_{D_{GE}} &= \frac{\partial C_D}{\partial h}(h - h_{GE}), \qquad h < h_{GE} \qquad (1) \\
\Delta C_{m_{GE}} &= \frac{\partial C_m}{\partial h}(h - h_{GE}), \qquad h < h_{GE}
\end{aligned}$$

The derivatives for the ground effect were estimated for each flap position η_K separately. Fig. 2 shows the identified coefficients plotted as a function of CG altitude/wing span (h/b). The maximum altitude for ground effect was identified to be $h/b = 0.5$ $(h_{GE} = 20\,m)$. Although no ground effect changes of the lateral directional motion could be identified, this modeling proved to be sufficient for landing simulation according to Level D requirements. For takeoff simulation, especially for aircraft rotation, an altered elevator effectiveness in ground effect was found. This increased effectiveness was modeled as a function of tail altitude above ground and elevator deflection rate and was added to the pitching moment equation. It may be pointed out here, although discussed later, that wind gradients affect the landing simulation fidelity considerably, and have to be modeled properly for the identification of the ground effect from flight test data.

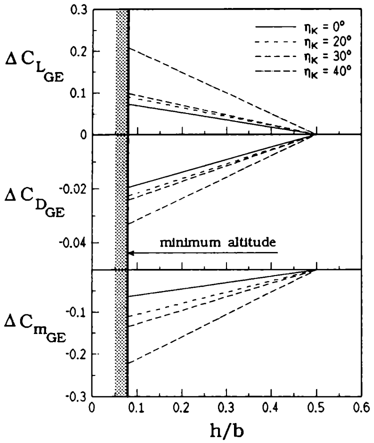


Fig. 2. Identified C-160 ground effect

Ground Handling Model

The ground handling model has to provide the gear forces and moments acting on the aircraft at and about its center of gravity. The ground handling model of the C-160 transport aircraft was formulated analytically and describes the dynamics of the nose gear and the four main gears separately[13].

The model includes two suspension devices for each single landing gear causing dynamics between the airframe and the runway (Fig. 3):

- The oleo-pneumatic damper is mounted with an inclination angle of about 40° to the fuselage reference line. Due to this inclination, a braking force results in a damper compression as well as a geodetic z-force. The kinematic model between the tire axes and the airframe uses the geometric lengths, angles of inclination and damper strokes of the struts. The oleo-pneumatic damper force F_d is the sum of damper stiffness and damping force. The damper stiffness F_s of the struts is a nonlinear function of damper compression and is known from manufacturer's data. The damping is modeled velocity squared $(\dot{f}^2_{oleo})$ with different constants for compression (K_1) and decompression (K_2):

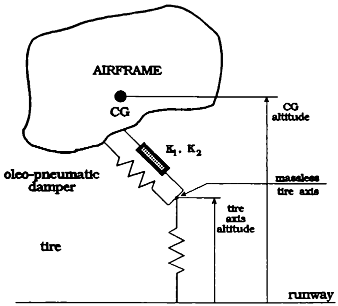


Fig. 3. Model structure of one C-160 landing gear

$$F_d = \begin{cases} F_s + K_1\,\dot{f}^2_{oleo} & : \ compression \\ F_s - K_2\,\dot{f}^2_{oleo} & : \ decompression \end{cases} \qquad (2)$$

- The tire is the second suspension device. Two tires are mounted on each strut. The nonlinear tire stiffness is modeled using manufacturer's data tables. A tire damping force is neglected as well as the tire mass. The geodetic z-motion of the tire axis for each strut is described as a first order differential equation.

For taxiing, a realistic model of the nose wheel side force as a function of the nose wheel skid angle is important. The C-160 tire side force model, which was analytically derived based on truck tire measurements, was adapted to C-160 taxi tests. These tests included a couple of rate of turn applications in the region of full adhesion and partial adhesion with slip, symmetric, and asymmetric braking up to 80 *kt* ground speed. The side force model is a function of the tire skid angle and the normal force, but also of aircraft ground speed and runway condition (dry, wet). The skid angle τ is, as illustrated in Fig. 4 for the nosewheel, the difference between the steering angle η and nosewheel drift angle:

$$\tau = \eta - atan\left(\frac{v_K + r \cdot x_{CG}}{u_K}\right) \qquad (3)$$

with r being the yaw rate, u_K, v_K being body-fixed components of the flight path velocity and x_{CG} being the distance between the CG and the tire contact point in body-fixed x-direction. The nonlinear tire side force model parameters were identified only for a dry concrete runway, as no taxi tests were conducted at wet or icy surface conditions. The identification result is shown in Fig. 5, in comparison to the analytical model. Since the identified tire side force as a function of skid angle differs only slightly from truck tires, the model parameters for other runway consistencies are taken from the analytical model. For illustration, in Fig. 5 the (analytical) side force is also plotted for wet runway condition. It is obvious, that its maximum value is decreased considerably and the nonlinear skid angle dependency begins already at small skid angles ($\tau < 4°$).

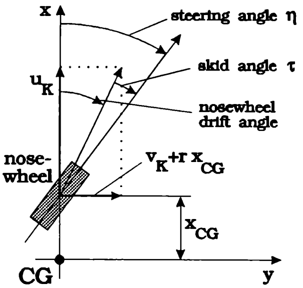


Fig. 4. Derivation of the tire skid angle

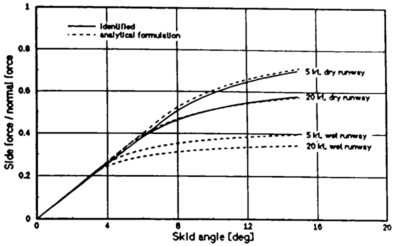


Fig. 5. Identified / analytical tire side force modeling

Validation and Simulation Problems on and Close to the Ground

Simulating the aircraft takeoff and landing, the identified airborne aerodynamic model is to be applied to flight conditions outside of the normal flight range. There are a few aspects that have to be considered.

The thrust is either extremely high (takeoff power) or extremely low or even negative (flight idle power for approach). The maximum takeoff thrust of each engine is about 70,000 N, whereas for flight conditions used for basic aerodynamic model identification its maximums are 23,000 N for 0° flap position and 40,000 N for 60° flap position. All aerodynamic derivatives accounting for thrust effects have to be checked especially for takeoff, whether they are valid at this much higher thrust level. Based on a two-point aerodynamic model formulation, the pitching moment is modeled as a function of horizontal tail lift multiplied by the tail lenght l_t. For tail lift computation, the effective angle of attack at the tail has to be computed, which is a function of the downwash at the horizontal tail ε_H:

$$\varepsilon_H = \frac{\partial \varepsilon_H}{\partial \alpha}\,\alpha\,(t - \tau_\alpha) - \frac{\partial \varepsilon_H}{\partial C_S}\,C_S\,(t - \tau_{C_S}) \qquad (4)$$

where the downwash lags are defined by $\tau_\alpha = l_t/V$ and $\tau_{C_S} = l_e/V$, with l_e being the distance between the tail aerodynamic center and the thrust effective point and

V being the true airspeed. C_S is the thrust coefficient: $C_S = T/2\bar{q}S_P$, with T = thrust, $\bar{q}$ = dynamic pressure and S_P = propeller area. This modeling leads to difficulties in the case of takeoff, since

$$\lim_{\bar{q}\to 0} C_S \to \infty. \qquad (5)$$

In order to avoid an unrealistic downwash angle, it is necessary to limit the thrust coefficient to $0 < C_S < 3$ for the C-160.

A second problem - in the C-160 case - especially for the landing approach is the thrust computation. For the engine model the measured torque, low pressure shaft speed and acceleration, and turbine gas temperature are used in order to compute the actual thrust as a sum of propeller thrust and net thrust of the turbine. For very low or even negative torques at flight idle the torque measurement is not valid, and one has to switch to an alternate thrust computation as a function of power lever position and true airspeed only. This switching may cause some abrupt thrust changes often just in the sensitive region shortly before touch down.

A third problem is the quality of the flow sensor data on and close to the ground. Angle of attack, angle of sideslip, and dynamic pressure are measured by a five-hole probe mounted on a rather short noseboom. These flow sensors were calibrated for thrust levels valid within the normal flight envelope. Analysing the approaches including touch down, it became obvious, that the angle of attack measurement is affected strongly by the ground effect, and the dynamic pressure by the thrust level, especially at reverse thrust or ground idle at high speed taxi. The validation of a landing demands a critical look at the flow sensor signals, since they are sometimes not suitable for verification of simulation quality, as will be shown later.

Validation Examples

The following examples are some snapshots out of the math model validation procedure with flight/taxi test data. A total sum of 62 maneuvers has to be documented within the whole operational envelope, including 23 maneuvers for aircraft dynamics with ground contact.

The FAA quality requirements are postulated in form of tolerances[1]. The proof-of-match compares the flight measured variable ± these tolerances with the model output. For Level D standard, some typical tolerances are: ground distance (±5% in time and distance), turn rate ($\pm 2°/sec$), airspeed ($\pm 3\,kt$), pitch angle and angle of attack (±1.5°), altitude ($\pm 6\,m$), roll and sideslip angle (±2°). Not all criterions are required in each maneuver.

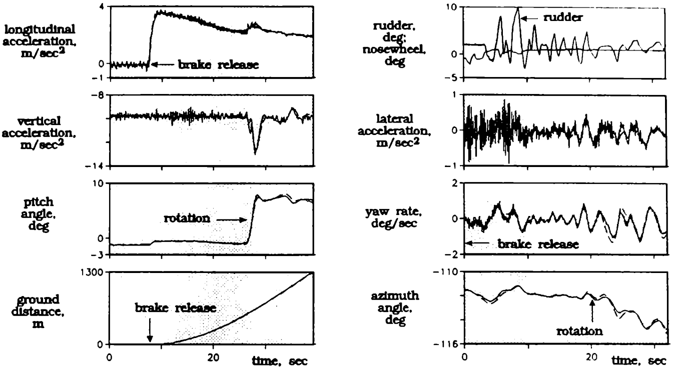
The validation simulation was included into a parameter estimation procedure in a similar way the aerodynamic model was estimated. But now, only the initial conditions of the state equations, some sensor offsets, and a wind profile are estimated, in order to validate the math model with maneuvers not used for aerodynamic model identification.

Normal takeoff

A typical normal takeoff validation with 20° flap deflection is shown in Fig. 6a for the longitudinal motion. Brake release results in a slight pitch angle increase, which can be seen in both measured and simulated data. Acceleration and ground distance are simulated very accurately, the position measurement of the aircraft was provided by a laser tracker. The roll friction coefficient is assumed to be 0.03, exactly the value given by the C-160 handbooks for takeoff on dry concrete. The difference between measured and computed ground distance is about 6 m at the end, that is 0.46 % of the takeoff distance. The maximum difference between measured and simulated pitch angle is 0.5° (maximum allowable for Level D: ±1.5°). A directional control simulation, although not Level D required, is presented in Fig. 6b for a 0° flaps takeoff. Nosewheel and rudder deflections, being responsible for directional control, are presented. The curve fit is quite good, the maximal difference in azimuth is 0.3°.

Critical engine failure on takeoff

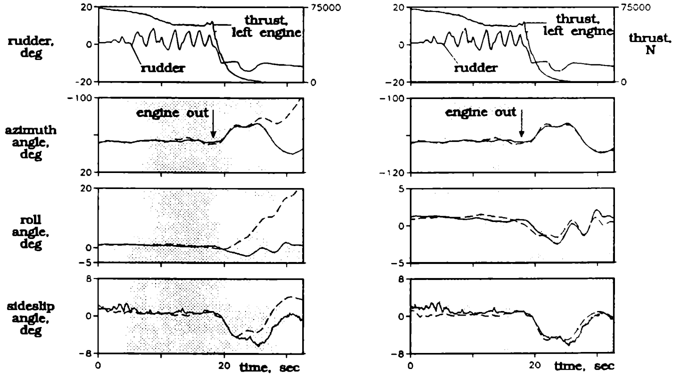
Problems may arise simulating an asymmetric thrust condition on takeoff. Fig. 7 presents a C-160 one engine out on takeoff validation for a 20° flap deflection. 18 sec after brake release the left engine, the critical one, is cut off. The pilot reacts and tries to maintain the azimuth using the rudder. In Fig. 7a aerodynamic effects due to asymmetric thrust are neglected. This modeling prooved to be only sufficient, if the flap position is 0°. For the landing flaps being deflected as in Fig. 7, a strong rolling moment is induced. It can be explained by the change of the thrust vector direction, as the propeller flow hits the deflected flaps. This can be modeled as an aerodynamic effect identifying an additional rolling derivative $C_{l_{\Delta T}}$ from single engine flight maneuvers. Also there is a different drag increase on the left and right wing resulting in a yawing moment. This was modeled by identifying an additional yawing derivative $C_{n_{\Delta T}}$. Additionally, it was found that the weathercock stability is influenced by asym-



(a) longitudinal, 20 deg flaps

(b) directional control, 0 deg flaps

Fig. 6. Normal takeoff validation (——— flight measured; - - - - simulated)



(a) neglecting asymmetric thrust influence on aerodynamics

(b) modeling asymmetric thrust influence on aerodynamics

Fig. 7. Critical engine failure on takeoff validation: influence of asymmetric thrust derivatives (——— flight measured; - - - - simulated)

metric thrust, which is modeled as $C_{n_{\beta_{\Delta T}}}$-derivative.

$$\begin{aligned} \Delta C_{l_{\Delta T}} &= (T_{left} - T_{right}) \cdot C_{l_{\Delta T}} \qquad (6) \\ \Delta C_{n_{\Delta T}} &= (T_{left} - T_{right}) \cdot \left(C_{n_{\Delta T}} + C_{n_{\beta_{\Delta T}}} \cdot \beta\right) \end{aligned}$$

Fig. 7b shows the simulation improvement for the critical engine out on takeoff accounting for these asymmetric thrust influences identified from single engine flight maneuvers. The FAA required accuracy for the roll angle and the angle of sideslip is well reached now (tolerance: ±2°).

Normal landing: influence of wind profile

Landing simulation, Level D required from 200 *ft* altitude down to nosewheel touch down, is a quite sensitive task, where a lot of physical effects have to be considered. First it is essential to set or estimate accurate initial conditions at the beginning of the simulation in 200 *ft* altitude above ground. In simulation, it is important to meet the exact aircraft attitude at touch down, as this is influencing the ongoing simulation in a quite sensitive and nonlinear way. If it is not possible to succeed, FAA allows to split up the landing validation into two segments: one for the approach and one for the derotation after touch down.

In addition to the ground effect influence, the modeling of wind and wind gradients proved to be a fundamental factor for the quality of landing approach simulation in order to meet the exact touch down point. The influence of an altitude dependent horizontal wind is demonstrated in Fig. 8. Fig. 8a shows the best curve fit that can be obtained estimating a constant wind from 70 *m* altitude ($\approx 200ft$) down to touch down. There are considerable deficiencies in pitch angle, indicated airspeed, angle of attack, and radio altitude. It is impossible to get the exact touch down point. Fig. 8b shows the landing simulation if a constant wind and two wind gradients, splitted into north (uw_g) and east components (vw_g), are estimated as a function of altitude h. In this example, the kink altitude for the wind gradients is set to 35 *m*.

$$\begin{aligned} uw_g &= uw_{g35} + \frac{\partial uw_g}{\partial h}(h-35) \\ vw_g &= vw_{g35} + \frac{\partial vw_g}{\partial h}(h-35) \end{aligned}$$

$$\ldots\ldots \begin{cases} h \leq 35\,m: & \frac{\partial uw_g}{\partial h} = \frac{\partial uw_{g1}}{\partial h};\ \frac{\partial vw_g}{\partial h} = \frac{\partial vw_{g1}}{\partial h} \\ h > 35\,m: & \frac{\partial uw_g}{\partial h} = \frac{\partial uw_{g2}}{\partial h};\ \frac{\partial vw_g}{\partial h} = \frac{\partial vw_{g2}}{\partial h} \end{cases} \qquad (7)$$

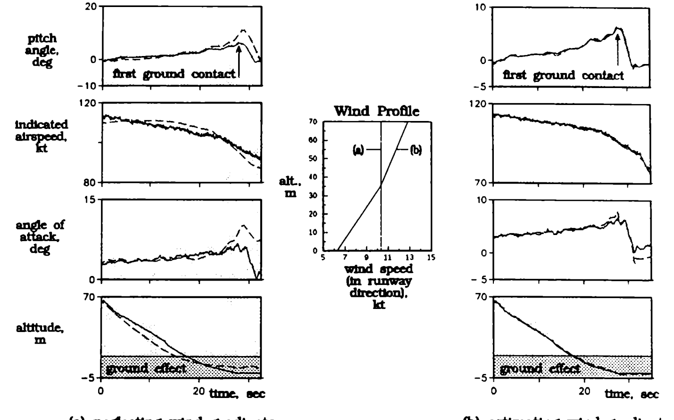
Fig. 8 presents the estimated wind profile in runway direction. Two wind gradients are found to be sufficient for landing simulation from 70 *m* altitude for this wind situation (for difficult wind situations more wind gradients may be necessary). Now it is possible to meet the exact touch down point and the landing simulation is sufficient for Level D requirements without splitting it into two segments.

As mentioned before, the angle of attack measurement of the five-hole probe is strongly affected by the ground effect. This can be seen in Fig. 8b. Down to 10 *m* radio altitude there is hardly any difference between measured and computed angle of attack. Below 10 *m* altitude the measurement error increases up to the maximal value of 2.5°, when the aircraft is rolling on the runway after derotation. The measurement of the indicated airspeed from the dynamic pressure measurement of the five-hole probe is not affected.

Improvement of touch down derotation

A second example of C-160 landing simulation improvement is shown in Fig. 9, which presents the time segment from main gear to nose gear touch down (aircraft derotation). The measured longitudinal acceleration shows a considerable peek at main gear touch down, which originates from the force needed to accelerate the eight wheels of the main undercarriage. In Fig. 9a, this tire acceleration force is neglected in the simulation model, and an obvious deficiency in pitch angle can be seen, which is violating the FAA Level D requirement for the pitch angle accuracy (±1.5°).

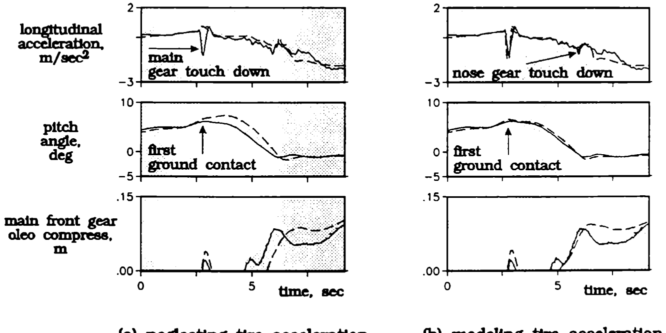
The geodetic x-force F_x at each wheel contact point can be computed if the geodectic z-force F_z and the actual friction coefficient μ are known: $F_x = \mu \cdot F_z$. But the friction coefficient between tire and runway surface changes in a strong nonlinear way from total slip to full adhesion. A simplified approach is adapted: the typical touch down impulse I of the main undercarriage $I = F_x \cdot t$ was estimated from the measured longitudinal acceleration a_x and the known aircraft mass m: $F_x = m \cdot a_x$, evaluating several landing impacts. The equivalent impulse time was identified to $0.2\,sec$, the corresponding force for each of the ten wheels to $\approx 6{,}000\,N$. An aircraft ground speed dependency was neglected. Fig. 9b shows the simulation fidelity using this touch down impulse modeling. The fit of longitudinal acceleration and pitch angle (the wheel forces of the main undercarriage have a considerable lever arm to CG) is considerably improved and now within the FAA Level D tolerances. The correct touch down point simulation of the main undercarriage can be seen, and even the nosewheel touch down after about 6 *sec* can be



(a) neglecting wind gradients

(b) estimating wind gradients

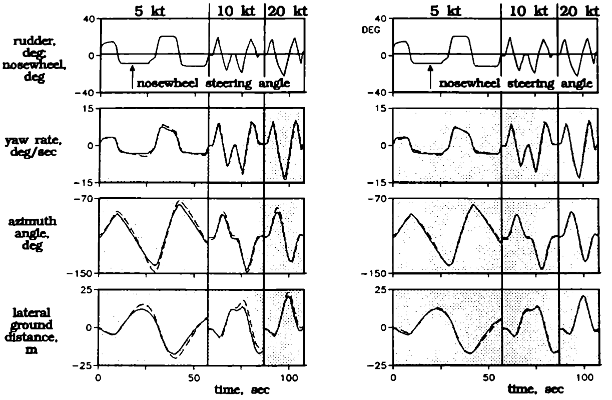
Fig. 8. Normal landing validation: influence of wind profile consideration
(——— flight measured; - - - - simulated)



(a) neglecting tire acceleration

(b) modeling tire acceleration

Fig. 9. Touch down derotation: influence of tire acceleration
(——— flight measured; - - - - simulated)



(a) simplified skid angle modeling

(b) precise skid angle modeling

Fig. 10. Rate of turn vs. nosewheel steering validation: influence of precise skid angle modeling (——— flight measured; - - - - simulated)

detected in both the measured and the computed acceleration. Furthermore, the touch down point is confirmed by the oleo compress plot of the main front gear. This oleo compress simulation, which is an important variable within the undercarriage force computation, shows an obvious improvement, too.

Influence of skid angle modeling on ground handling

The tire skid angle (cf. Fig. 4) at each gear strut is an important parameter for the ground handling model side force computation. Eq. (3) describes the nose gear skid angle as the difference between the steering angle and the drift angle at the tire contact point. As the main gear struts are not steerable and mounted near the CG (neglecting the influence of the lever arms), a simplified tire skid angle at the main gear struts can be defined: $\tau = -atan(v_K/u_K)$, which is exactly the aircraft drift angle at CG. Fig. 10 (a) shows the curve fit in a rate of turn vs. nosewheel steering angle validation test for 5, 10, and 20 *kt* ground speed using this simplification. It is obvious that there are deficiencies, especially for low speed taxi with 5 and 10 *kt*. The drift angle at each main gear contact point depends on the lever arm in body-fixed x-direction as well as in y-direction, if there is a yaw rate. Considering these lever arms, the extended formulation of the main gear tire skid angle is now:

$$\tau = -atan\left(\frac{v_K + r \cdot x_{CG}}{u_K - r \cdot y_{CG}}\right) \qquad (8)$$

with yaw rate r, the lever arms from the CG to the tire contact points with the runway x_{CG}, y_{CG}, and the body-fixed components of the flight path speed u_K, v_K. Fig. 10b shows the curve fit using this precise skid angle computation for the tires of the main undercarriage. There is a considerable improvement and the low speed taxi is modeled properly now.

If there are large changes in the yaw rate r combined with low taxi speeds, the term $u_K - r \cdot y_{CG}$ is extremely important and can decide about the sign of the ground speed at a main gear tire contact point. One typical example is the minimum radius turn validation test, where the ground speed of the inner main gears may be near zero.

Conclusions

The validation of a C-160 simulator math model identified from flight data demanded some extended modeling on and close to the ground in order to satisfy the FAA postulated Level D standard. If deficiencies between simulated and measured data are detected, it is essential to separate aerodynamic, thrust, undercarriage, and atmospheric effects in order to update the right model part. The takeoff and especially the landing validation demands some experience as there are not only a lot of nonlinear effects - for example the additional gear strut forces from the landing impact - but also the distorted flow sensor signals in ground effect and at low speed taxi. Several C-160 validation examples show the improvement in simulation quality updating different math model parts: the aerodynamic model for one engine out takeoff, the ground handling model for derotation after touch down, the tire skid angle modeling for taxi handling, and the wind situation influence on landing simulation accounting not only for a constant wind but for a wind profile.

Acknowledgements

The authors gratefully acknowledge the support provided by the German Air Force, the German Procurement Organisation (BWB), and the German Air Force Flight Test Center WTD-61, which all helped to carry out the C-160 Data Gathering Project in a short span of about a year. The authors are also grateful to the German Air Force and the simulator manufacturer Thomson-CSF for having granted the permission to publish the investigations reported in this paper.

References

1. FAA; *Airplane Simulator Qualification.* FAA Advisory Circular, AC 120-40B, Interim Version, Feb. 1991.

2. Neville, K. W. and Stephens, A. T.; *Flight Update of Aerodynamic Math Model.* AIAA Atmospheric Flight Mechanics Conference, Paper No. 93-3596, Monterey, CA, Aug. 1993.

3. Maine, R. E. and Iliff, K. W.; *Identification of Dynamic Systems - Application to Aircraft. Part 1: The Output Error Approach.* AGARD-AG-300, Vol. 3, Dec. 1986.

4. Allen, L. D.; *Evolution in Flight Simulation.* AIAA Atmospheric Flight Mechanics Conference, Paper No. 93-3545, Monterey, CA, Aug. 1993.

5. Baillie, S. W., Hui, K., and DeLeeuw, J.; *The Flight Test and Data Analysis Program for the Development of a Boeing/deHavilland Dash 8 Simulator Model.* 80th Symposium of the AGARD Flight Mechanics Panel on "Flight Testing", Paper No. 30, Crete, Greece, May 1992.

6. Krag, B., Jategaonkar, R. V., Mönnich, W., and Fischenberg, D.; *Estimation of an Aerodynamic Data Base for a New C-160 "Transall" Flight Simulator from Flight Data.* RAeS Symposium on "Data Issues for Flight Simulators - An On-going Problem", Paper No. 7, London, UK, Nov. 1993.

7. Neville, K. W.; *Simulator-to-Flight Validation - Methods, Problems and Dilemmas.* RAeS Symposium on "Data Issues for Flight Simulators - An On-going Problem", Paper No. 14, London, UK, Nov. 1993.

8. Kullar, I.; *Simulation Data Issues for Large Civil Transport Airplanes.* RAeS Symposium on "Data Issues for Flight Simulators - An On-going Problem", Paper No. 2, London, UK, Nov. 1993.

9. Jategaonkar, R. V., Mönnich, W., Fischenberg, D., and Krag, B.; *Identification of C-160 simulator data base from flight data.* 10th IFAC Symposium on System Identification, Copenhagen, Denmark, July 1994.

10. Jategaonkar, R. V., Mönnich, W., Fischenberg, D., and Krag, B.; *Identification of Speed Brake, Ramp Door, Air-Drop, and Landing Gear Effects from "Transall" Flight Data.* AIAA Atmospheric Flight Mechanics Conference, Paper No. 94-3473, Scottsdale, AZ, Aug. 1994 (to be published).

11. Etkin, B.; *Dynamics of Atmospheric Flight.* John Wiley & Sons, Inc. NY, 1972.

12. Staufenbiel, R., Yeh, B.; *Flugeigenschaften in der Längsbewegung von Bodeneffekt-Fluggeräten.* ZFW 24, 1976, page 3-9 and 65-70.

13. Fischenberg, D.; *Identification and Validation of a C-160 Ground Handling Model.* DLR IB 111-94/07, Feb. 1994.

AIAA-94-3405-CP

OPEN AND CLOSED LOOP CONTROL WITH A PERSPECTIVE TUNNEL-IN-THE-SKY DISPLAY

E.Theunissen[*] and M.Mulder[**]
Delft University of Technology
Mekelweg 4
P.O. Box 5031
2600 GA Delft, The Netherlands

Abstract

It is generally accepted that the spatial presentation of the desired flightpath in an ego-centered reference frame can be used to improve spatial and navigational awareness. However, this is not the only advantage of perspective flightpath displays. In contrast to the closed-loop error correcting control strategy typical for flight director command displays, the trajectory preview available with Tunnel-in-the-Sky displays allows the pilot to use an error neglecting control strategy and exercise a certain amount of open-loop control, while the range of error gains resulting from the perspective projection allows adaptable closed-loop control. To determine the relations between trajectory preview and the different control strategies, two experiments will be conducted. In the first experiment the relation between the initiation of a control action and a parameter introduced here as the Time-to-Wall Crossing will be determined. The second experiment serves to investigate whether the information required for the estimation of the Time-to-Wall Crossing is obtained through trajectory preview.

Introduction

It is generally accepted that the spatial presentation of the desired flightpath in an ego-centered reference frame can be used to improve spatial and navigational awareness[18]. However, this is not the only advantage of perspective flightpath displays.

When flying a perspective flightpath display, the pilot can use the information about his current position and heading relative to the flightpath for closed-loop control. The dimension of the tunnel determines the error gain with which position errors are presented to the pilot, and thus influences his control behaviour. Furthermore, the tunnel dimension and the number of cross section frames influence the perceived velocity of the observer.

In 1993 an experiment was conducted to determine pilot performance and control behaviour when flying a Tunnel-in-the-Sky with the addition of a flightpath vector and with a position predictor for different error gains[19]. It was found that in the flightpath vector configuration tracking accuracy increased linearly with decreasing tunnel size, and control activity was linearly related to error gain. In the predictor configuration no significant difference in control behaviour was found for the different tunnel dimensions. This suggests that in the presence of an adequate prediction of the future position and attitude the pilot does not use the error information presented by the tunnel, but only the error presented by the predictor for his closed-loop control task. The increasing lateral and vertical accuracy with decreasing tunnel size is explained by the fact that since the reference square of the predictor reduces, the pilot's ability to estimate the center of the square increases. As a result of these findings, one might conclude that for the guidance task it suffices to present a predictor with a reference frame, and skip the tunnel.

However, it must be taken into account that the tunnel provides the pilot with a high level of spatial awareness and with trajectory preview information. The question rises how the pilot uses this preview information. Does he use the guidance cues presented by the tunnel for his inner-loop control actions, does he use the tunnel to determine whether a deviation from the desired position as indicated by the predictor needs correction, or does the tunnel only provide him with spatial awareness?

Pilot control behaviour is determined by an integration of a number of parameters, i.e. Cross Track Error (XTE), Track Angle Error (TAE), Track Angle Error Rate (TAER), Vertical Error (VE) and Flightpath Angle Error (FPAE). It would be of great advantage if a single parameter could be identified, which can be used to predict pilot control actions.

In this paper, the Time-to-Wall Crossing (TWC) parameter is introduced as such a parameter. The TWC represents a prediction of the remaining time before the aircraft crosses one of the tunnel walls, and is based on the Time-to-Line Crossing (TLC) parameter introduced by Godthelp[3] during his research into car driver control behaviour. It is hypothesized that with a perspective flightpath display the TWC parameter can be used to predict the initiation of open-loop control actions and the initiation of corrective actions during error neglecting control.

To determine whether a relation between the TWC and the initiation of open-loop and error correcting control actions can be found, an experiment in the moving base flight simulator at the Faculty of Aerospace Engineering of Delft University of Technology will be conducted. To verify whether this information is obtained through trajectory preview, a second experiment will be conducted in which this preview is varied by manipulating the visible part of the tunnel.

[*] PhD candidate, Electrical Engineering

[**] PhD candidate, Aerospace Engineering

Background and related work

In this section, specific properties of a perspective flightpath display are discussed in the context of findings from related research. Most aircraft related research into visual cues addresses the naturally available cues. Since accurate position cues are only dominantly available during the landing phase or low-level flight, these are the two major areas covered by this research. At higher altitudes, only rotations can be perceived with a high enough resolution.

Since the visual cues available from a perspective flightpath display are not naturally available in the aircraft environment, and the required computer performance for artificially generating them was prohibitive until the early eighties, research on this specific topic is relatively scarce in the aerospace community.

In contrast, with car driving these cues are continuously available. Ample research has been performed on the subject and the established framework with respect to the perception and processing of this information and the resulting control can be translated to the aircraft situation and used to predict pilot usage of these cues. In the following sections, results from research on perception of the environment, effects from a limited field of view, the perception of velocity, the effects of scene content on altitude and glideslope control, preview, and control behaviour are discussed in the context of perspective flightpath displays.

Perception of the environment

Gibson[1] hypothesizes that the focus of expansion is used in car driving by keeping it in the direction the vehicle must go. Gordon[4] discusses the principles applying to the perception of positional, velocity, and acceleration fields under rectilinear and curvilinear motion. He states that since experimental evidence suggests that drivers guide themselves by reference to the road edges and the center stripe, the often quoted statement that the focus of radial outflow is the cue for the direction of sensed locomotion is challenged. Although the direction of vehicle motion is related to the focus of expansion, the focus itself is not an effective cue. He further motivates this by arguing that the focus of expansion of a flat horizontal plane lies at the vanishing point in the sky or will occupy points on trees or buildings if the road is curved. Generally, it is difficult, if not impossible, for the driver to locate the focus of expansion, and contrary to Gibson[2], the borders and lane markings are used in vehicular guidance. When the vehicle is aligned with a straight or regularly curved highway, the road assumes a steady state appearance. The borders and lane markers remain almost stationary in the driver's field of view. A road of constant curvature assumes a steady state appearance up to the break in curvature. Lateral guidance could consequently be considered to involve the maintenance, through visual feedback, of an acceptable steady state condition and nulling the deviations from it[4].

If the moving vehicle is misaligned laterally with the road, the entire field of view moves as a unit. Riemersma[17] investigated the optical cues in the dynamic visual field which are related to the movements of the vehicle. He distinguishes between two components for the control of lateral position: lateral motion and heading rate. The heading rate reflects itself in an optical translation of all points of the visual field. The lateral motion becomes optically manifest as rotations of the optical images of the edgelines, without a change in position of the optical vanishing point. The extent of lateral misalignment is indicated by the rate of extent of slewing and side-slipping of the road borders and lane markers.

With the Tunnel-in-the-Sky, this information is presented by the lines parallel to the desired trajectory. In case the XTE, TAE, VE, and FPAE are all equal to zero, the tunnel is displayed as a symmetrical shape. Consequently, with the Tunnel-in-the-Sky a deviation from the desired trajectory is perceived through an asymmetry in the presentation. If any of the four parameters is unequal to zero, the symmetry disappears. When referring to the deviation of a tunnel element from its position in the symmetrical condition as the displayed error, the contribution of XTE, TAE, VE, and FPAE can be expressed as a function of the relative position of the tunnel element, the Field-Of-View (FOV) and the screensize. Equation 1 presents the contribution of the XTE to the error presented on the display (x_{err1}), and Equation 2 presents the contribution of the TAE (x_{err2}). The sum of x_{err1} and x_{err2} is the error presented on the display.

$$x_{err1}=\frac{sc}{2\tan(fov/2)}\frac{XTE}{d} \qquad (1)$$

$$x_{err2}=\frac{sc}{2\tan(fov/2)}(\tan(atan(\frac{w/2+XTE}{d})+TAE\ )-\frac{w/2+XTE}{d}) \qquad (2)$$

In both equations *sc* represents the screensize in meters, *fov* the field of view in degrees, and *d* the distance in meters from the viewpoint to the point in the 3-D world which is used as the reference for the XTE and TAE. Note that from Equation 2 it follows that the contribution of the TAE on x_{err2} depends on the current XTE, and becomes negligible for a large distance between the reference point and the viewpoint (Equation 3).

$$\lim_{d\to\infty} x_{err2}=\frac{sc}{2\tan(fov/2)}\tan(TAE\ ) \qquad (3)$$

As a result, the observer can use the section of the tunnel at a large distance from the viewpoint to estimate the TAE, and the nearby section for the XTE. Figure 1 illustrates the trajectory with a preview of 200 m in the absence of errors. Figure 4 illustrates the same trajectory at a preview distance between 500 m and 1000 m. Figure 2 illustrates the nearby trajectory preview with zero TAE and an XTE which is 25% of the tunnel width and Figure 3 illustrates the nearby trajectory preview with a TAE of 5 degrees and zero XTE. Figure 5 illustrates the situation of Figure 2 at a larger distance from the viewpoint, and Figure 6 illustrates the situation of Figure 3 with preview between 500 m and 1000 m. The asymmetry in the vertical dimension is caused by the fact that the pictures have been calculated for an attitude aligned viewing vector. In order to follow the flightpath, the velocity vector must be aligned with the direction of the flightpath.

When XTE is replaced with VE, and TAE with FPAE, the equations present the relations for the vertical dimension. It must be noted however, that these equations are not valid for curved segments, and that the accuracy of the TAE thus depends on the distance to the next curved segment. Also, it must be stressed that the goal of presenting these equations is not to indicate how the observer processes the information from the display, but only that the information is present and due to the familiarity of the observer with the 3-D world probably will be processed in a way which allows him to make separate estimates of XTE, TAE, VE, and FPAE. As a result of this ability the pilot will be better able to update his internal representation of the dynamics of the system under control. Finally, it is important to realize that the amount of spatial trajectory preview can both be used to improve the accuracy with which parameters such as current TAE can be estimated (Equation 3) and to anticipate the future forcing function. Grunwald[5] illustrated that with a perspective image of a future circular vehicle path the error to be zeroed can be expressed as Equation 4,

$$\varepsilon = \frac{1}{2}\nu + \chi + \frac{\eta}{D} \tag{4}$$

with D the looking distance,
ν the angle of D along the future circular vehicle path,
χ the current TAE, and
μ the current XTE.
Grunwald[5] approximates Equation 4 as:

$$\varepsilon = \frac{1}{2}\frac{D}{V^2}\ddot{\eta} + \frac{1}{V}\dot{\eta} + \frac{1}{D}\eta \tag{5}$$

with V representing velocity.

This form of equation is often referred to as a quickened display[18] of the parameter η, which is comparable to the algorithm driving a flight director in which the XTE has to be nulled. When taking only one preview distance D into account, the information presented is basically the same. Conventional flight directors are based on a weighted combination of position- and angular errors and error rates. In the horizontal dimension, XTE and TAE are used to calculate the deflection of the vertical flight director bar. In the vertical dimension, FPAE and VE are used to calculate the deflection of the horizontal flight director bar. As a result of the integration of multiple parameters into a single dimension, the pilot is unable to synthesize information about the specific errors from the flight director display. Furthermore, since the error gains of the display are determined by the flight director algorithms, the possible bandwidth the pilot can apply for scanning and executing the flight director commands is very limited. In situations where the required performance is less than the performance for which the gains have been determined, the pilot is forced to maintain the higher gain. Due to the absence of preview on the future forcing function the pilot is forced to continuously apply an error correcting control strategy. The trajectory preview available with navigation displays in today's glass cockpits could provide the pilot with some information about when to expect a flight director command for a turn initiation. However, due to the constant scaling of the future trajectory, these display formats do not present the trajectory preview in a way which allows the pilot to adapt his gains during a flight director tracking task. In contrast, the perspective projection employed with the Tunnel-in-the-Sky display allows the presentation of information with a much broader range of error gains. As a result, the spectrum of possible gains which the pilot can apply is more extensive, he can use an error neglecting control strategy, and as a result it is possible to choose an optimum between workload and performance.

Effects of the limited field of view

One of the fundamental differences between the available visual information with car driving and the information with perspective guidance displays is the rather limited field of view of the latter ones. To compensate for the resulting missing cues, a variety of display augmentation concepts are possible. Grunwald[5] investigated the effectiveness of three different basic display augmentation concepts for guidance of low flying Remotely Piloted Vehicles (RPVs). His results show a strong dependence of the effectiveness of the display aids on vehicle dynamics and the spectrum of disturbances.
In his research into perspective display formats for the presentation of guidance information, Grunwald[5,6] indicates that the lack of cues which result from the narrow field of view can yield an undamped system. He proposes the use of predictive display symbology to compensate for these missing cues. Another reason to present the pilot with predictive symbology is to allow him to better determine the moment an anticipatory control action is required for curve initiation. In this way, the required closed-loop control behaviour after the open-loop action is reduced and performance is increased.

Perception of velocity

Gordon[4] discusses three mechanisms for the perception of motion. At slow rates, motion is inferred from a change in position, at more rapid rates motion is directly perceived, and at still more rapid rates, motion appears as blur. With the tunnel display speed cues are presented through the motion of the cross section frames toward the observer. The velocity gain is defined as the velocity of an element on the display divided by the relative velocity between the observer and the element[19]. Equation 6 presents the expression for the velocity gain V_g.

$$V_g = \frac{sc\ \tan(\frac{fov}{2})}{tunnelsize} \tag{6}$$

with *sc* the screensize in meters, *fov* the field of view, and *tunnelsize* the dimension of the tunnel in meters.
It is unlikely that in the presence of such compelling integrated speed cueing, the pilot does not use this information. Thus, velocity as perceived from the motion of

the tunnel elements will probably influence pilot control behaviour, e.g. the moment of the initiation of a control action to enter a curve. In the context of the TWC concept, where the pilot is supposed to estimate the time before the tunnel is crossed from the visual scene, it is apparent that the estimate requires an indication of velocity from this scene. This does not make the speed indicator superfluous. On the contrary, for velocity control the pilot will probably prefer the accurate presentation of this variable in a single spatial dimension. Given that the cues provided by the motion of the perspective flightpath are adequate, this redundancy in the presentation of velocity also provides an additional alerting mechanism for changes in velocity.

Two important speed cues available in the 3-D world are optical edge rate and global optical flow rate. Optical edge rate is defined as the speed at which texture elements pass a given point in the subject's field of view[20]. As indicated by Larish[10], the term edge rate is misleading as the actual information appears to be local average texture flow rate as noted in Warren's[20] definition, rather than actual edges. Optical flow rate of texture elements within the visual field is determined by the following equation[2]:

$$\mathit{Flowrate} = \frac{V}{h}\,\sin^2(\mathit{elevation})\cos(\mathit{azimuth}) \qquad (7)$$

where *elevation* and *azimuth* refer to the optical coordinates of a texture element and *V/h* is a global scaling factor determined by the speed *V* and the altitude *h* of the observer. Gordon[4] indicated this relationship by stating that 'Distance to the surface must be specified to permit a judgement to be made of the speed of motion'. Warren[20] named this global scaling factor optical flow rate. He hypothesized that the perception of egospeed would scale with global optical flow rate.

For rectilinear flight over a flat surface, edge rate is completely determined by the speed of the observer and the spacing of the edges on the surface. Edge rate is independent of altitude. Therefore, edge rate is an accurate indicator of ground velocity when altitude varies, but not when ground texture varies. With a Tunnel-in-the-Sky display, optical edge rate is determined by the distance between the successive frames and increases with increasing distance from the center of the screen. Global optical flow rate is defined as the rate of expansion of the visual field, which is a ratio of forward velocity and altitude. Therefore, flow rate is a reliable indicator of ground velocity only under the condition of constant altitude. In case of a perspective flightpath display, global optical flow rate is determined by the geometric field of view and the tunnel size.

Larish[10] examined the relative contribution of optical edge rate and global optical flow rate to the perception of egospeed under viewing conditions in which the degree of three-dimensional cueing was varied. In the uncontrolled condition, subjects monitored a conventional display with a perspective representation of a moving 3-D scene. In controlled viewing conditions, stronger 3-D cues where presented through the use of a binocular device. He reports that optical edge rate appears to be an important variable in the perception of egospeed under all conditions. Optical edge rate was the dominant source of information used in the uncontrolled conditions, with global optical flow failing to contribute significantly. Under controlled viewing conditions, a much greater effect of global optical flow rate was reported. He hypothesized that the result is due to the ability to perceive depth in the controlled viewing conditions. The effect of global optical flow rate might become even more pronounced if stronger 3-D information is provided. The results suggest that the cues used in the judgement of egospeed change as a function of the availability of conflicting 2-D depth cues. Since the current implementation of our Tunnel-in-the-Sky display is a perspective presentation on a conventional 2-D display device, it is expected that the perception of egospeed is mostly determined by optical edge rate.

Johnson[7] conducted an experiment to determine the ability to control groundspeed in the presence of relevant and irrelevant variations in edge and flow rates. He tested whether more experienced pilots would be better able to ignore irrelevant variations in edge rate, and whether pilots would show a bias toward using edge rate relative to global optical flow rate for the control of ground velocity. He reports that no evidence was found that people are more intrinsically sensitive to edge rate variation, nor that pilots may be biased toward using edge rate to control ground speed. As indicated previously, it is anticipated that with the tunnel display pilots use the separate airspeed indicator for velocity control.

Effects of scene contents on altitude and glide slope control

Research on this topic is often performed in the context of the required realism of Computer Image Generators (CIGs) for Flight Simulators. With CIG, often a trade-off is required between the number of objects in a scene and the level of detail of these objects.

Kleiss[9] evaluated whether the apparent size of more detailed and familiar appearing objects serves as an additional cue for altitude control in simulated low-level flight. Results showed no difference between abstract objects and familiar objects. However, performance did improve with increasing object density. His results suggest that CIG processing capacity may be most effectively utilized by increasing object density rather than individual object detail.

This supports Wilckens'[21] hypothesis that when enough cues are available, the human observer completes the rest of the picture. In other words, the level of realism of an object must exceed a certain threshold after which it does not significantly contribute to altitude control. These findings justify the rather basic representation of the future desired flightpath and the runway as employed in our display format. When developing more advanced 3-D display formats to obtain a Synthetic Vision System (SVS), this fact should also be taken into account.

A possible cue for glide slope control during landing is the angle between the aimpoint and the horizon, sometimes referred to as the H-angle. Lintern[11] showed that distortion of this angle by simulation of up-sloping or down-sloping terrain beyond the runway influenced glide-slope control in a predictable way. He proposes the use of texture lines parallel to the runway centerline to allow the pilot to estimate the real horizon, and thus obtain an accurate estimate of the H-angle. With our Tunnel-in-the-Sky display we overlay the heading tape over the real horizon (Figures 1-6), so the real H-angle is always directly visible, also in the presence of up-sloping terrain.

Preview

Gordon[4] poses that 'Perceptual anticipation is of central importance to the driving task. The driver must anticipate at least one reaction time ahead if he is to meet the current situation'. The driver's visual fixation distance has been related to anticipation requirements.

Wohl[22] believes that fixation distance D may be predicted by $D=\tau*V$, where τ is the driver's response lag and V vehicle velocity. If the driver would not look at least this far ahead, he could not respond appropriately. However, it is generally accepted[3,4,5] that the driver does not view a fixed distance ahead, and this model is too much of a simplification.

Gordon[4] describes the typical scanning behaviour of a driver as follows: 'he looks far ahead, returns to a middle distance, and seemingly in disregard of anticipation requirements, he may check his alignment with the road and nearby vehicles'. With the tunnel display this corresponds to looking far ahead to estimate the TAE and anticipate changes in the trajectory which require future action, looking at an intermediate distance to determine a possible excursion of the constraints and anticipate changes in the trajectory which require almost immediate action, and looking very nearby to estimate the current XTE. As indicated previously, the presence of a range of error gains allows the operator to select his own weighting function, and choose to neglect errors and error rates within certain constraints. Thus, the availability of trajectory preview allows an error neglecting control strategy to be applied.

Control behaviour

Ample research has been performed on human control behaviour in compensatory and tracking tasks[8,12,13,14]. However, research into pilot control behaviour when presented with the information typical for perspective flightpath display formats is relatively scarce. An extensive literature review about the modelling of pilot control behaviour with spatial displays is presented by Mulder[16].

Again, with car driving the situation is different. Various models have been proposed to describe the driver control behaviour in relation with the visual environment. McRuer[15] presents an approach in which he distinguishes between compensatory, pursuit and dual mode control behaviour. With compensatory control, the driver uses lateral position and heading errors. With pursuit control the driver takes advantage of the trajectory preview to initiate an open-loop control action to follow the desired path, i.e. the driver applies feedforward control. With dual mode behaviour, the driver initiates an open-loop control action which is succeeded by closed-loop compensatory control.

It is hypothesized that the latter description is representative for pilot control behaviour when the perspective flightpath display presents an approaching curve. With the Tunnel-in-the-Sky display, the future XTE is a function of the current XTE and the TAE. If the current XTE is acceptable, the value of the TAE determines when the pilot will intervene. This control strategy, which is only possible with preview on the future forcing function, is referred to as error neglecting. A large TAE will cause the XTE to increase more rapidly than a small TAE. As indicated in the previous section, accurate information about the TAE is obtained from tunnel elements at a larger distance from the viewpoint. Thus, the accuracy with which the TAE can be estimated depends on the amount of trajectory preview which is presented. To predict the future error when flying a straight tunnel section, pilots probably use the TAE and make a linear prediction. When approaching a curve, the pilot will use the trajectory preview to initiate an open-loop control action to obtain the desired bank angle. Since no reference for the correct track angle is available in a curve, it is much harder to estimate the TAE and future path errors. It is expected that the pilot will switch to compensatory control by making use of the information about the XTE and the trend in the XTE to maintain a stationary condition. As indicated by Equation 2, this information is best available at a short distance from the viewpoint (further away from the center of the screen).

A possible control strategy would be to keep the intersection of the tunnel walls with the screen boundary at a constant position. In this way, the pilot zeros the trend in the XTE, which in turn is identical to zeroing the TAE. Another strategy would be to allow a certain trend in the XTE, as long as it is below a certain threshold. Intervention will take place when the XTE or the trend in the XTE exceeds a certain threshold. This discussion already illustrates that the control strategy of the pilot may depend upon the value of a number of parameters. Summarizing, with the Tunnel-in-the-Sky display the trajectory preview allows a certain amount of open-loop control to be applied and the range of gains resulting from the perspective presentation allows adaptable closed-loop control.

Time-to-Line Crossing

Gordon[4] states that 'The behaviour involved in steering an automobile, for instance, has usually been misunderstood. It is less a matter of aligning the car with the road than it is a matter of keeping the focus of expansion in the direction one must go'. The velocity field provides information on the speed and direction of the vehicle's forward motion. The driver may become aware of the misalignment of the car by slewing shifts in direction, and by side-slipping sidewise movements which exceed the human visual position and movement thresholds. The driver's perceptual response is based upon an integration of these and other information.

On the basis of human perception theory, it is difficult to determine which of the four combinations of slew, sideslip, rate, and amplitude the driver perceives. The driver responds to a total situation, not to isolated or ranked cues. This indicates the necessity of determining a single parameter to describe and predict driver responses. Godthelp[3] introduced the so-called Time-to-Line Crossing concept, which is based on the assumption that there is a relation between the remaining time the vehicle under control is within a certain boundary, and the moment a control action is initiated.

Most of the available vehicle control models are based on the fundamental assumption that drivers control their vehicle with permanent visual feedback. However, as it is commonly accepted, visual feedback is sometimes interrupted. Godthelp[3] investigated the potential role of visually open-loop strategies and error neglection in vehicle control. He assumes that the time available for a driver to control his vehicle in an open-loop mode largely depends on the accuracy of the open-loop generated steering-wheel action and the time available for error neglection.

The control activity indicates the amount of effort invested in the control task. For continuous closed-loop control tasks, frequency domain techniques are very useful for describing control behaviour. However, for non-continuous control behaviour encountered during error neglection and open-loop control, time domain techniques may be more appropriate. Godthelp[3] conducted several car-driving experiments in which he evaluated the concept of the TLC. His results indicated that:

1. Drivers are very capable to estimate the required amount of open-loop control to initiate a curve.
2. The accuracy of an open-loop control action to initiate a curve decreases with increasing curvature.
3. The necessity for compensatory control increases with increasing curvature.
4. The TLC after the initiation of the open-loop control action decreases with increasing curvature.

To gain more insight in the control behaviour of drivers in the temporary absence of visual information, Godthelp[3] studied the occlusion strategies adopted by drivers. His results indicated that the occlusion time, i.e. the time the driver is willing to control his vehicle in the absence of visual information, is approximately 40% of the total TLC. When drivers were told to ignore position and heading errors, and only apply control at the moment they think is necessary in order to remain on the road, drivers adopted a strategy with a fixed TLC. From this, Godthelp[3] concluded that the control strategy is strongly determined by the degree of uncertainty about the future vehicle trajectory.

TLC and perspective displays

In this paper it is hypothesized that the answer to the question of how the pilot uses the preview presented by the tunnel depends on the task he is confronted with. When he is told to fly as accurate as possible, he will use the information with the highest error gain he can process to perform his task. In case of an additional flight director or predictor, the pilot will mainly concentrate on the information presented by this indicator. On the other hand, when his task is to keep the error below the thresholds indicated by the walls of the tunnel, he can apply a much wider variety of control strategies.

Thus, in contrast to the flight director task where the pilot functions as an error correcting mechanism, the flexibility of Tunnel-in-the-Sky display allows the pilot to select a task dependent optimal control strategy. This flexibility is illustrated in Figure 7. The lower line presents the results from an experiment[19] in which the pilots were told to fly as accurate as possible. It indicates the maximum accuracy which can be achieved as a function of the tunnel size. A limit for the maximum error when remaining in the tunnel is presented by the upper line, which indicates half the tunnel size.

Figure 8 presents the aileron control activity for the experiment mentioned above. As can be seen, control activity decreases with increasing tunnel size. It is expected that when the task of the pilot changes from flying as accurate as possible to remaining within the tunnel, control activity will also decrease. The theoretical minimum control activity is independent of the tunnel size, and mainly determined by aircraft velocity, the curvature of the trajectory and disturbances. The lower line indicates the theoretical minimum for a certain trajectory. To achieve an accuracy which lies between te lines of Figure 7, the pilot can adopt a control strategy which is comparable to the dual mode behaviour discussed by McRuer[15].

At Delft University of Technology it is being investigated whether the TLC concept can be used to predict the moment when a pilot will perform a corrective control action. Also, it will be examined whether the TLC parameter can be used to predict the moment of initiation of an open-loop control action for curve entrance and exit. Finally, the relation between curvature, accuracy of the initial control action and the resulting compensatory control behaviour will be examined. Based on these findings, it will be investigated whether the TLC concept can be used to determine a minimum and maximum required preview distance.

Since the TLC concept was defined for a 2-D problem, it had to be changed to a 3-D equivalent, and from now on we will refer to the time parameter as the Time-to-Wall Crossing. Based on the current state of the aircraft, a prediction is made about its future trajectory. When assuming that during the prediction interval velocity and yaw rate remain constant, the future path of the aircraft can be described as a circle segment. On a straight section Equation 8 presents the estimated TWC as a function of cross track error, track angle error, velocity, yaw rate and tunnel width in the horizontal plane:

$$TWC = \frac{\cos^{-1}\left(\cos(TAE) - \frac{r\left(\frac{width}{2} - XTE\right)}{V}\right)}{r} + \frac{TAE}{r} \qquad (8)$$

In Equation 8 *width* represents the tunnel width in meters, *XTE* the cross track error in meters, *TAE* the track angle error, *V* the velocity in m/s and *r* the yaw rate in radians/sec.

Due to the contribution of the yaw rate in the prediction of the TWC, this parameter correlates better with the aileron deflections than the TAE.

Experiment

An experiment will be conducted to verify the hypothesis made about the TWC. For this purpose, pilots will fly a curved approach with a Tunnel-in-the-Sky display.

Since the flightpath consists of straight and curved sections, the algorithm implemented to estimate the TWC is a little different from Equation 8. A number of position predictions are performed, the intersection of this trajectory with the tunnel boundary is calculated, and an estimate of the TWC is computed.

This is performed in the horizontal and the vertical dimension. In this way it is possible to correlate horizontal TWC values with aileron control activity, and vertical TWC values with elevator control. The hypothesis is that the moment the pilot initiates a corrective action is determined by this TWC. If this is the case, the pilot should have an internal representation of the system which allows him to predict the TWC. The introduction of the TWC implies that an assumption is made regarding the internal representation of the pilot. Based on the previous research performed into car-driving[3,15], it is assumed that the pilot uses the current position error, the heading error (which amounts to the rate in the current position error), and the rate of the heading error, which on a straight section is equal to the yaw rate. The results of this experiment should give an indication whether the TWC parameter can be used for the same purpose as its two-dimensional equivalent with car driving.

To check whether the pilot uses the preview information to estimate the TWC, a second experiment will be conducted in which the preview is less than the velocity multiplied by the average TWC.

Description of the experiment

For the experiment, the selection of the tunnel width is an important parameter. If the tunnel is too small, the pilot will apply mostly closed-loop control behaviour. This makes it very difficult to determine the initiation of the open-loop control action, and thus makes it hard to determine the relation between the initiation of the control action and the remaining TWC. It is expected that when the tunnelsize exceeds a certain threshold, the TWC is no longer a function of the tunnel dimension. Only the time between two control actions will be longer for a larger tunnel, since it takes more time to reach a tunnel wall.

We use six different flight plans in the experiment. Each flight plan consists of two straight and one curved segment. Each flight starts at an altitude of 2000ft. The length of the final straight segment is 6300 m, and the curve is either +45 degrees or -45 degrees. Godthelp[3] reports that with car-driving the closed-loop control behaviour after the initiation of the curve increases with increasing curvature. To verify whether this relation is also true with the tunnel display, we include three different curvatures, corresponding to 10, 15, and 20 degrees nominal bank angle at a velocity of 120 knots. The respective curve radii are 1008, 1370, and 2081 m.

The straight sections are used to determine whether a relation can be found between the TWC and the moment the pilot initiates a control action. The curved sections are used to check whether the TWC parameter is better than either XTE or TAE to describe control strategy. To prevent dominantly closed-loop control behaviour, the pilot is informed that the goal of the experiment is not to determine the maximum accuracy which can be achieved.

In the second experiment the amount of trajectory preview presented is equal to the product of the average TWC of experiment one and aircraft velocity. Equation 2 indicates that this reduces the ability of the pilot to estimate the TAE. It is hypothesized that for curve entrance and curve following the control behaviour remains the same, since the TAE is not the dominant source of information. However, during the straight section the TLC is expected to reduce due to the decreased accuracy with which the TAE can be estimated. This is based on Godthelp's hypothesis[3] that the control strategy depends on the degree of uncertainty of the vehicles future trajectory.

An increase in noise in the presentation of the aircraft position and attitude will result in a decrease in TWC. This is important since in an aircraft the perspective image is calculated from a viewpoint determined by the positioning system, and a viewing direction determined by the heading and the attitude determination system. It is important to consider that the output of both systems is always accompanied with noise, which introduces noise in the presentation of the perspective flightpath.

Results

The main experiments will start in June '94. Preliminary evaluations revealed that as a result of the flexibility offered by the display format, pilots can and will apply different control strategies. With error neglecting control, the basic strategy is to make a corrective action with the goal to avoid the imminent boundary excursion. A second goal can be to maximize the time to the next required control action. The success of this approach depends on the degree of certainty with which the future flightpath can be predicted.

Results of these preliminary evaluations indicate that the different control strategies might be identified from plots of the TWC, the XTE, and the TAE. Figure 9 shows an example of a plot of the TWC and the aileron deflections, Figure 10 of the XTE and the aileron deflections, and Figure 11 of the TAE and the aileron deflections for a situation in which the pilot corrects for the imminent boundary excursion and tries to maximize the time until the next control action. Figures 12-14 show plots of respectively TWC, XTE, and TAE for a situation in which the pilot only wants to avoid the current boundary excursion. The plots show that both TAE and TWC give good indications of the moment the pilot initiates a corrective action. However, the TWC better correlates with the subsequent control actions, which is primarily due to the contribution of the yaw rate in the TWC. It must be stressed that these preliminary results do not contain any statistic significance, since the main experiment still has to be performed. The results from this experiment will be presented at the conference.

Conclusion

With perspective flightpath displays, the trajectory preview allows an error neglecting control strategy and a certain amount of open-loop control to be applied. The range of error gains resulting from the perspective projection allows adaptable closed-loop control. Based on research conducted in the field of car driving, a time domain parameter, the so-called Time-to-Wall Crossing parameter has been introduced for the description of open-loop control actions and error neglecting control. Results from a preliminary evaluation confirm the usefulness of the Time-to-Wall Crossing parameter. To prove the statistical significance of this parameter in relation to the description of open-loop and error neglecting control, an experiment will be conducted in June '94.

References

1. Gibson, J.J. *The Perception of the Visual World* (1950), Houghton, Mifflin, Boston.

2. Gibson, J.J., Olum, P., Rosenblatt, F. 'Parallax and perspective during aircraft landings' *American Journal of Psychology* (1955), 68, pp. 372-385.

3. Godthelp, H. 'Studies on Human Vehicle Control' *PhD Thesis* (1984) Institute for Perception TNO, Soesterberg, The Netherlands.

4. Gordon, D.A. 'Perceptual Basis of Vehicular Guidance' *Public Roads* (August 1966), Vol. 34, No. 3, pp. 53-68.

5. Grunwald, A.J., Merhav, S.J. 'Effectiveness of Basic Display Augmentation in Vehicular Control by Visual Field Cues' *IEEE Transactions on Systems, Man, and Cybernetics* (September 1978), Vol. SMC-8, No. 9, pp. 679-690.

6. Grunwald, A.J., Robertson, J.B., Hatfield, J.J. 'Experimental Evaluation of a Perspective Tunnel Display for Three-Dimensional Helicopter Approaches' *AIAA Journal of Guidance and Control* (Nov.-Dec. 1981), Vol. 4, No. 6, pp. 623-631.

7. Johnson, W.W., Awe, C.A. 'Use of Optical Edge and Optical Flow Rate Information in the Perception and Control of Ground Velocity' *Proceedings of the Seventh International Symposium on Aviation Psychology* (1993), pp. 286-291.

8. Kleinman, D.L., Baron, S., Levison, W.H. 'An Optimal Control Model of Human Response Part I: Theory and Validation' *Automatica* (1970), Vol. 6, pp.357-369.

9. Kleiss, J.A.. Curry, D.G., Hubbard, D.C. 'Effect of Three-Dimensional Object Type and Density in Simulated Low-Level Flight' *Proceedings of the Human Factors Society - 32nd Annual Meeting* (1988), pp. 1299-1303.

10. Larish, J.F., Flach, J.M. 'Judgement of Speed with Computer Generated Motion Displays' *Proceedings of the Fourth International Symposium on Aviation Psychology* (1987), pp. 244-250.

11. Lintern, G. and Liu, Y.T. 'Explicit and Implicit Horizons for Simulated Landing Approaches' *Human Factors*, Vol. 33, No. 4 (1991), pp. 401-417.

12. McRuer, D.T., Graham, D., Krendal, E., Reisener, W. 'Human Pilot Dynamics in Compensatory Systems - Theory, Models, and Experiments with controlled element and Forcing Function Variations' *AFFDL-TR-65-15* (1965).

13. McRuer, D.T., Jex, H. 'A Review of Quasi-Linear Pilot Models' *IEEE Transactions on Human Factors in Electronics* (September 1967), Vol. HFE-8, No.3, pp. 231-249.

14. McRuer, D.T., Weir, D.H., Klein, R.H. 'A Pilot-Vehicle Systems Approach to Longitudinal Flight Director Design' *AIAA Journal of Aircraft* (1971).

15. McRuer, D.T., Allen, R.W., Weir, D.H., Klein, R.H. 'New Results in Driver Steering Control' *Human Factors* (1977), Vol. 19, No. 4, pp. 381-397.

16. Mulder, M. 'Displays, Perception and Aircraft Control - A Survey of Theory and Modelling of Pilot Behaviour with Spatial Instruments' *Report LR-762* (April 1994), Delft University of Technology, The Netherlands.

17. Riemersma, J.B.J. 'Perceptual Cues in Vehicle Guidance on a Straight Road' *Proceedings of the 2nd European Annual Conference on Human Decision Making and Manual Control* (1982), pp. 127-136, Bonn, Germany.

18. Stokes, A.D., Wickens, C.D., and Kite, K. *Display Technology - Human Factors Concepts* SAE 1990.

19. Theunissen, E. 'A Primary Flight Display for Four-Dimensional Guidance and Navigation: Influence of Tunnel Size and Level of Additional Information on Pilot Performance and Control Behaviour' AIAA-93-3570-CP. *Proceedings of the AIAA Flight Simulation Technologies Conference* (1993), pp. 140-146. Monterey, CA.

20. Warren, R. 'Optical transformation during movement: Review of the optical concomitants of egomotion' *Technical Report AFOSR-TR-82-1028* (1982). Bolling AFB, DC: Air Force Office of Scientific Research (NTIS no. AD-A122 275).

21. Wilckens, V. and Schattenmann, W. 'Test Results with New Analog Displays for All-Weather Landings'. *Problems of the Cockpit Environment* (1968), AGARD CP No. 55.

22. Wohl, J.G. 'Man-Machine Steering Dynamics' *Human Factors* (1961), Vol. 3, pp. 222-228.

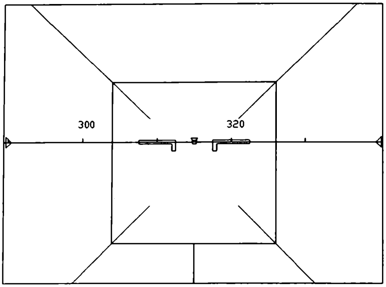


Figure 1 XTE=0, TAE=0, preview 200m

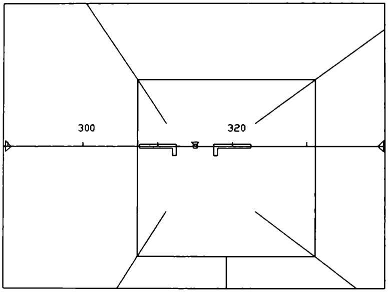


Figure 2 XTE=25% width, TAE=0, preview=200m

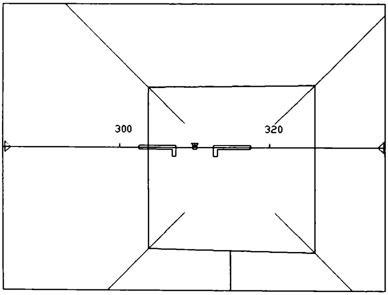


Figure 3 XTE=0, TAE=5, preview=200m

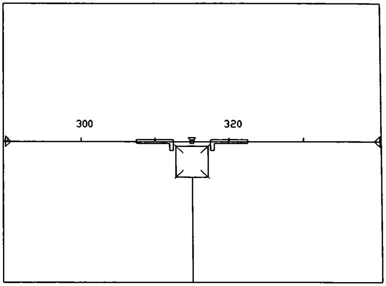


Figure 4 XTE=0, TAE=0, preview=500-1000m

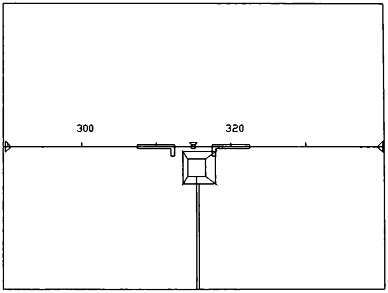


Figure 5 XTE=25% width, TAE=0, preview=500-1000m

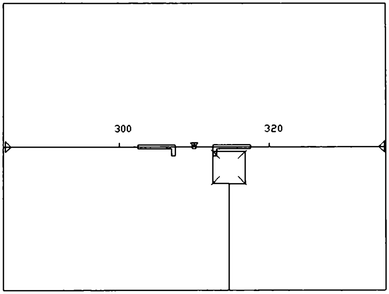


Figure 6 XTE=0, TAE=5, preview=500-1000m

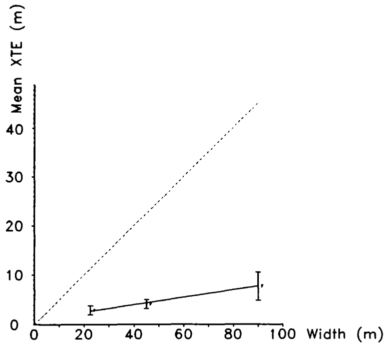


Figure 7 Position accuracy versus tunnel size for the FPV configuration

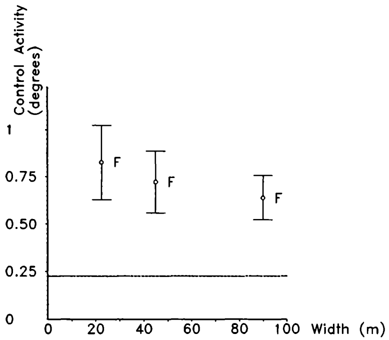


Figure 8 Control activity versus tunnel size for the FPV configuration

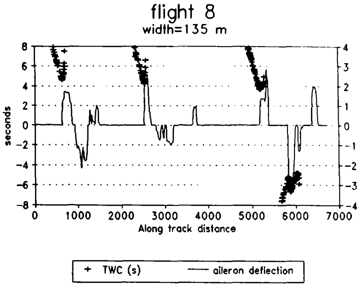


Figure 9 TWC and aileron deflection as a function of along track distance

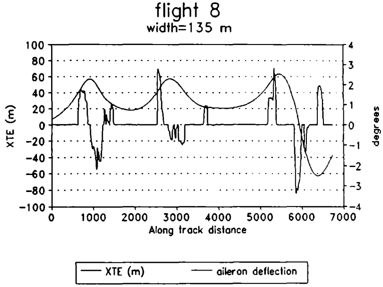


Figure 10 XTE and aileron deflection as a function of along track distance

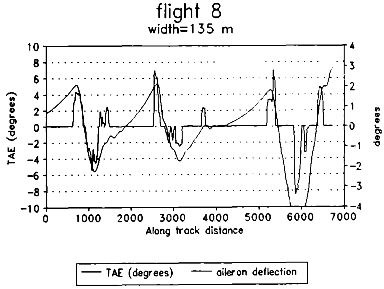


Figure 11 TAE and aileron deflection as a function of along track distance

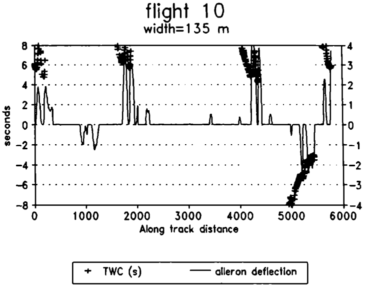


Figure 12 TWC and aileron deflection as a function of along track distance

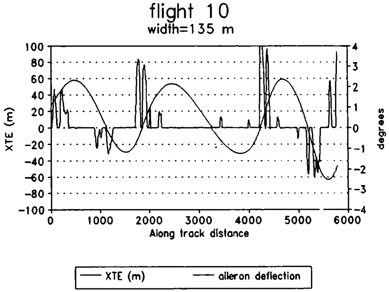


Figure 13 XTE and aileron deflection as a function of along track distance·

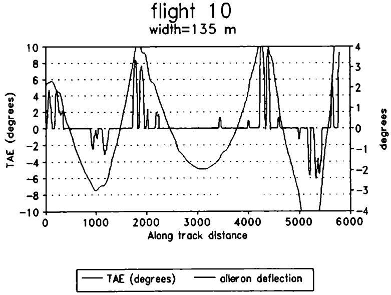


Figure 14 TAE and aileron deflection as a function of along track distance.

CHOICE AND JUDGMENT: THE FUTURE OF FLIGHT TRAINING

Stephen C. Hayne
Faculty of Management, MIS
University of Calgary
Calgary, Alberta
schayne@acs.ucalgary.ca

C.A.P. Smith
School of Business Administration
University of Montana
Missoula, Montana
cap@selway.umt.edu

ABSTRACT

Improvements in pilot judgment training have the potential to save hundreds of lives every year. Based on the Applied Learning Theory and the Recognition-Primed Decision model, we introduce an inexpensive computer simulator that incorporates actual flight video segments to provide pilots with judgment training. Emphasis is place on making choices because research shows a potential benefit of framing decisions in this way. Furthermore, the simulator incorporates an element of time pressure to make the training more realistic. Practice with realistic scenarios and continued reinforcement of correct behavior will hopefully lead to improved judgment training for general aviation pilots

INTRODUCTION

The aviation community has been investigating the process of judgment training for some time. In a study by Jensen and Benel[9], it was reported that 52% of all fatal accidents in general aviation between 1970 and 1974 could be attributed to poor judgment. Jensen and Benel concluded that improvements in pilot judgment could be obtained through training.

The issue of judgment training was investigated further by Berlin, Gruber, Holmes, Jensen, Lau, Mills, and O'Kane[1]. In this study, Berlin et al. identified five hazardous thought patterns that may precede poor pilot judgment: Anti-authority, External control, Impulsivity, Invulnerability, and Machismo. Anti-authority is the tendency to disregard rules and procedures. External control refers to the attitude that the pilot has little or no control over their fate. Impulsivity refers to a decision process in which pilots perform the first action that comes to mind. Invulnerability is an attitude in which a pilot is convinced that nothing bad can happen to him or her. Machismo refers to a tendency of pilots to attempt difficult or dangerous tasks for the purpose of gaining admiration. Each of these thought patterns has been associated with aviation accidents. Berlin et al. conclude that pilot judgment training should be designed to reduce the incidence of these hazardous thought patterns.

Numerous other studies of judgment training have been published in recent years[2,3,4,5,8,13,16]. The concept that a pilot's judgment is subject to improvement through training has become widely accepted. For example, many FAA safety seminars now include a reference to the five hazardous thought patterns.

In the aviation literature, the term "pilot judgment" has come to mean a "decision process done well." For example, Jensen and Benel define judgment in part as the ability to search for *relevant* information, generate alternative actions, and execute a *suitable* course of action[9]. This definition of judgment refers to both a sequence of actions and the quality with which those actions are performed. We prefer a more formal definition of judgment. We distinguish between two classes of decision-related tasks: choices, in which the decision maker selects one alternative from two or more, and judgments, in which he or she assesses the value of some state or outcome. For example, one judges the length of a runway or the strength of a wind; however, one chooses in which direction to land, or how much flap should be extended. Sometimes a choice can be made either directly (as in a decision as to which of two runways appear longer) or indirectly, based on judgment (as in first judging the length of each runway independently, then comparing them).

Judgment and choice are of course intimately connected, but the cognitive processes by which they are made are distinct[7,17] and may change in different ways in response to time pressure. A potential problem with traditional primary instruction is that almost all instruction is conducted without time pressure. The *judgments* that an instructor will emphasize on the ground do not always translate into correct *choices* when in the air.

Consider the following excerpt from a pilot who recently received his private license after accumulating barely 80 hours[10]. Already he has scared himself. It isn't that his flying is poor - his basic air work (e.g. stalls, coordinated turns, radio work and navigation) are all superior to what is prescribed in the Private Pilot Practical Test Standard. As he says, "It isn't my flying, it's my judgment! When on the ground, it's easy to think rationally. But when in the air..."

> *It all started with my first real flight on the 172. I took my room-mate and a friend up. I did the San Francisco East Bay Tour. I took off from Oakland, went to the Golden Gate Bridge, turned around, then headed to San Pablo Bay where I showed my passengers some of the maneuvers, then headed for Concord. After a can of coke and some rest, we went up again. Flew to the Altemont (sic), then near Livermore. At around that time, I noticed a thick overcast on the other side of the hills separating Oakland and Livermore. I got the Oakland ATIS which was reporting 2000 broken. It didn't look broken, it looked overcast, about 1100 bottom and 1500-2000 top. Well, I called up Bay Approach, and they asked me if I wanted an approach or VFR into Oakland. I answered that I was not instrument rated, and I am VFR into Oakland. They told me to stay out of Class B airspace and to navigate my own way to Oakland as needed to stay away from clouds. I elected to follow freeway 580 to freeway 80, then to Oakland next to the freeway. I started the descent and watched the altimeter wind down. I was then thinking, "What am I doing? Can I handle this? Sure I can. It's overcast, but under the clouds is decent visibility and I can follow the freeway. If I get an engine out, I'll land on the center divider or the numerous golf courses. I can do this....then again, may be I should go back to Livermore, and call and have my club pick me up...nah, I can do this." So there I was, barely 5 hours PIC and scud running!*

In the full account, which was posted to an Internet news group (rec.aviation.stories), the pilot exhibits a few of the hazardous thought patterns mentioned above. For example, the pilot exhibits the External Authority pattern when he "called up Bay Approach." It appears that he expected the controllers to tell him what actions to take, rather than merely supply information and clearances in response to his requests. Later, the pilot exhibits the Invulnerability pattern when he thinks, "...nah, I can do this."

A natural response for many pilots, upon hearing the real scenario above, is to claim "I would **never** have put myself in that situation," or "If I were in that situation, then I wouldn't have done what he did." Such statements, often made in the comfort of the pilot's lounge, reaffirm the speakers' beliefs in their own good judgment. However, once in the air, expressing good judgment as a series of good choices is not easy. We believe that there are two critical issues in the success of pilot judgment training that have received relatively little attention. The first issue is the presence of time pressure. Judgments made by pilots, especially during a critical in-flight event (CIFE), naturally involve some degree of time pressure. A primary difference between discussing a CIFE in the classroom, and dealing with one while in the air, is the presence of time pressure. The second important issue, we feel, is the way that problems are framed. In a comparison of choices and judgments, Smith[15] found that when decisions were framed as judgments concerning the utility of a single alternative, decision quality was substantially reduced as time pressure was increased. However, when decisions were framed as choices between alternatives, the decision quality was excellent, and remained almost unchanged as time pressure was increased. Based on these findings, we believe that pilot judgment training might benefit by framing decisions as choices, rather than judgments. This paper describes an architecture for a multi-media simulator designed explicitly to place pilots in time pressured choice situations.

LEARNING and MENTAL SIMULATION

Pilots must possess a variety of knowledge: theoretical knowledge such as an understanding of the principles of weather, factual knowledge such as the rules regarding airspace, and procedural knowledge such as how to control an airplane. In one way or another, new pilots must learn all of this knowledge. Given the amount of knowledge to be learned, pilot training programs rely heavily on learning theories to achieve some degree of efficiency.

Probably the most famous learning theory is the theory of Operant Conditioning. This theory is based on B. F. Skinner's[14] work with animals, and describes a way in which behavior is learned through repetition and reinforcement. A large variety of reinforcement schedules have been described in the literature, but all of them rely on repetition. It is through repetition that a student learns the consequences of the desired, and undesired, behaviors.

One of the distinguishing characteristics of pilot training is that there are a number of behaviors that must be learned without *any* repetition. For example, it is important that all pilots learn: never fly into a thunderstorm, never fly into a box canyon, and never fly VFR into IMC. The consequences of these behaviors are too dangerous to allow the student to learn directly. However, avoidance of these dangerous, unpracticed, behaviors is precisely what is meant when we speak of a pilot having "good" judgment. Thus for a pilot, good judgment by its nature should not be learned through repetition. An implication of this conclusion is that Operant Conditioning is an inappropriate theory on which to bas a pilot judgment training program. Nevertheless, pilots must gain experience at recognizing potentially dangerous situations.

The importance of being able to recognize potentially lethal situations has been studied by Klein and Crandall[11] in their work on mental simulation. In their study of urban fireground commanders (FGCs) making decisions in handling non-routine incidents during emergency events[12], they found that FGCs decision-making did **not** fit into a decision tree framework. They argued that they were not "considering alternatives" or "assessing probabilities"; rather they were acting and reacting on the basis of prior experience. Nonetheless, the FGCs were clearly encountering choice points during each incident. They were aware that different actions could be taken, but rarely thought about advantages or disadvantages of the actions. FGCs used their ability to recognize and classify a problem which would help them discover a "typical" way of reacting to it. They would use any available time to evaluate the course of action by applying mental simulation to "watch" the action unfold, to search for flaws and to discover what might go awry.

Klein and Crandall developed the Recognition-Primed Decision (RPD) model of rapid decision making to describe the process they observed. The RPD model (see Figure 1) is characterized by the following features:

- Situation recognition allows the decision maker to classify the task as familiar versus unfamiliar.

- Recognition as familiar includes this information: plausible goals, cues to monitor, expectancies and typical reactions.

- Options for courses of action are generated serially, with a very typical course of action as the first one considered.

- Option evaluation is performed serially, using mental simulation.

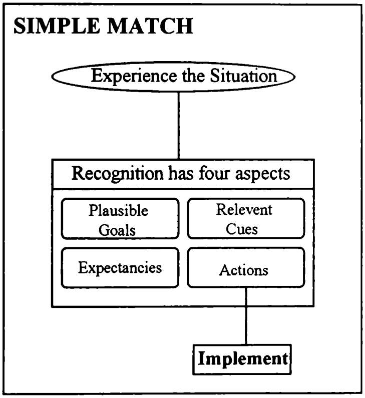


Figure 1: **Recognition-Primed Decision Model**

Klein and Crandall's RPD model points out the importance of experience in dealing with decisions made under time pressure. However, as mentioned earlier, the scenarios with which pilots need familiarity are often too dangerous to experience directly, as required by training programs based on Operant Conditioning. An alternative theory appropriate for pilot judgment training is the Applied Learning Theory[6]. This theory has three key elements: modeling, role playing, and transfer. Modeling refers to the provision of a role model for appropriate behavior. Role playing refers to the ability to simulate various scenarios in order to learn the consequences of different behaviors. Transfer of the desired knowledge is accomplished by providing explanations of appropriate theories, through practice with realistic scenarios, and continued reinforcement of correct behavior.

In the case of traditional pilot judgment training the model for behavior is generally a flight instructor. The flight instructor typically provides verbal simulations of potentially dangerous flight scenarios, and offers feedback to the student regarding the possible outcomes from their actions. Applied Learning Theory suggests that a more efficient transfer of knowledge will occur if the training is more realistic. The verbal role playing described above is less realistic than could be provided with a computer aided judgment simulator. Thus, according to Applied Learning Theory and Recognition-Primed Decision Model, a simulator should have an advantage over traditional pilot judgment training methods. The computer assisted simulation can be more effective than mental simulation because a larger set of variables and interactions can be introduced at differing rates. There are other potential advantages as well, e.g., it could be programmed to act as a role model (auto-pilot mode), and to provide consistent feedback for correct behavior (scenario replay and analysis).

In the following section we describe a computerized judgment simulator. Instructional aids of this type can be classified as either compensatory aids or training aids. A compensatory aid is one for which a benefit is derived only so long as the compensatory aid is in use. An example of a compensatory aid is the use of corrective lenses to improve vision. A training aid provides a benefit which persists after the aid is removed. Our judgment simulator is designed as a training aid. The judgment simulator is designed to allow role playing of potentially dangerous flight scenarios in a realistic fashion. In particular, the scenarios are presented with an element of time pressure, which as mentioned earlier is not normally associated with this type of training. Furthermore, the scenarios are presented in such a way that the students are presented with discrete choices, because our research suggests a potential benefit for framing decisions in this way.

ARCHITECTURE

We are targeting the design of our judgment simulator for equipment that is readily available and runs on personal computers (PCs). While the system architecture presented is platform independent (see Figure 2), PCs are becoming quite commonplace at flight training centers. It is our hope that by designing the system to perform well on a PC, it can be made available to a large number of pilots. A personal computer in the power range of a 486DX2/66, 8 MB of RAM, a large screen monitor, video connection to a controllable laser disk, game port and fast hard disk should be suitable. Using the Intel Smart Video Recorder, we have captured 22 flight segments, each ranging in length from 10 to 30 seconds. As well, 33 short clips of actual instruments were grabbed. The video clips were obtained from a library of broadcast quality in-flight footage belonging to Flyright Productions. For our prototype we chose the highest quality video (lowest compression) stored at a rate of 150kB/s for a total of 80 megabytes of video.

Like many PC simulators, the computer screen is divided into two parts; the scenery outside the cockpit and the instrument panel. Simulation would not be complete without audio. Sound is digitized and added from the sound library depending on the situation, i.e. other aircraft, passengers, engine(s), wind, air traffic control.

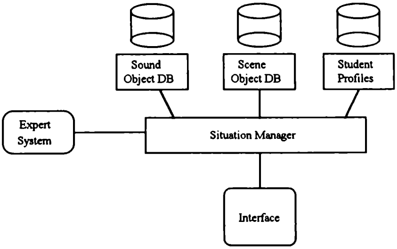


Figure 2: **Flight Simulator Architecture**

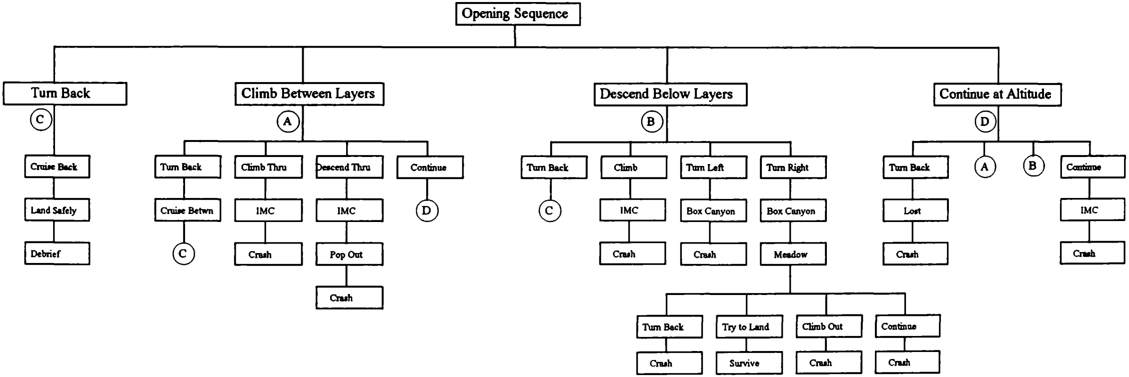


Figure 3: **Simulation Decision Tree**

This multi-media simulator explicitly places pilots in time-pressure, decision making situations. The architecture for this simulator involves scene and sound servers coupled with an expert system for decision management (see Figure 2). The situation manager controls the interface, while receiving inputs from the expert system and requesting objects from the sound and scene servers. The expert system has several major functions:

- using the student's historical performance data, the probability that certain events occur, will change
- initialization from a range of "easy" to "hard" impacts the choice of events and the degree of time pressure
- autopilot mode can be used to provide a role model of appropriate behavior
- scenario replay and analysis can be used to provide reinforcement of correct behavior.

By incorporating an individual pilot's past performance in new situations, e.g. if they have achieved poorly in an area, the system can continue presenting those situations. Furthermore, capturing performance is extremely useful for debriefing the flight. The decision scenario can be printed out and discussed with the student in order to determine the place where an error was made.

In the current prototype, events occur with a probability of 1.0 (see Figure 3) and time pressure limits are constant. As more experience is gained with managing video and more segments are added to the database, system choices will increase dramatically.

The prototype decision tree (see Figure 3) places the pilot in situations where choices need to be made; the airplane takes off on a cross-country flight through mountain terrain. The pilot is placed in straight and level cruise with deteriorating weather and at specified intervals the system presents a series of choices (press on, turn back, land, etc.). The pilot must choose one alternative within five seconds, or a default choice will be made for them. At each decision point, the specific choice made will determine which scenario will be presented next. Eventually, the choices lead to the conclusion of the simulated flight, either successfully as in a landing at an airport, or unsuccessfully as in a crash.

CONCLUSION

The approach we have taken to pilot training is well grounded in the latest theories of learning and decision-making during emergency situations. We have applied computer technology in an inexpensive simulator that has the benefit of realism and limitless geography/weather variation through the use of video. The pilot is placed in time pressured circumstances which require choices to be made. With repeated exposure to these situations, we believe that pilot judgment may improve. Ultimately we hope that the use of our system will help to save lives.

REFERENCES

[1] Berlin, J.I., E.V. Gruber, C.W. Holmes, P.K. Jensen, J.R. Lau, J.W. Mills, and J.M. O'Kane, (1982). *Pilot Judgment Training and Evaluation*, (DOT/FAA/CT-82/56), Daytona Beach, FL: Embry-Riddle Aeronautical University.

[2] Braun, R.J., and S.R. Trollip, (1982). "Towards and Internal Model in Pilot Training," *Aviation, Space, and Environmental Medicine*, 53(10):996-999.

[3] Brecke, F.H., (1982). "Instructional Design for Aircrew Judgment Training," *Aviation, Space, and Environmental Medicine*, 53(10):951-957.

[4] Buch, G., and A. Diehl, (1984). "An Investigation of the Effectiveness of Pilot Judgment Training," *Human Factors*, 26(5):557-564.

[5] Connolly, T.J., B.B. Blackwell, and L.F. Lester, (1989). "A Simulator-Based Approach to Training in Aeronautical Decision Making," *Aviation, Space, and Environmental Medicine*, 60:50-52.

[6] Goldstein, A.P. and M. Sorcher, (1974). Changing Supervisor Behavior, New York, Pergamon.

[7] Hogarth, R.M., (1981). "Beyond Discrete Biases: Functional and Dysfunctional Aspects of Judgmental Heuristics," *Psychological Bulletin*, 90(2):197-217.

[8] Jensen, R.S. (1984). "Pilot Judgment: Training and Evaluation," *Human Factors*, 24(1):61-73.

[9] Jensen, R.S., and R.A. Benel, (1977). *Judgment Evaluation and Instruction in Civil Pilot Training*, (FAA-RD-78-24). Savoy, IL: University of Illinois Aviation Research Laboratory.

[10] Karahashi, M. (1993). Excerpt from posting on *rec.aviation.stories*, Usenet.

[11] Klein, G. and B. Crandall, (1993). "The Role of Mental Simulation in Problem Solving and Decision Making". Flach, J.M., Hancock, P.A., Caird, J.K. and Vicente, K.J. (Eds.). An Ecological Approach to Human Machine Systems II: Local Applications. Hillsdale, NJ: Erlbaum.

[12] Klein, G., R. Calderwood, and A. Clinton-Cirocco (1986). "Rapid Decision Making on the Fire Ground," *Proceedings of the 30th Annual Human Factors Society*, 1:576-580.

[13] Lester, L. and D. Bombaci, (1984). "The Relationship between Personality and Irrational Judgment in Civil Pilots," *Human Factors*, 26(5):565-572.

[14] Skinner, B.F., (1953). Science and Human Behavior. New York, Macmillan.

[15] Smith, C.A.P., (1990). Decision Making Under Time Pressure: The Effects of Time Pressure on Information Search Strategy, Decision Strategy, Consistency, and Outcome Quality, Unpublished doctoral dissertation, University of Arizona.

[16] Stone, R.B., G.L. Babcock, and W.W. Edmunds, (1985). "Pilot Judgment: An Operational Viewpoint," *Aviation, Space, and Environmental Medicine*, 56:149-152.

[17] Yates, J.F. (1990). Judgment and Decision Making. Englewood Cliffs, NJ.

GENERAL HUMAN MACHINE INTERFACE REQUIREMENTS FOR ATM & AVIONICS SYSTEMS DESIGN

S. E. Gikas & P. Markopoulos
HCI Laboratory, Department of Computer Science
Queen Mary and Westfield College, University of London
Mile End Road, London E1 4NS
United Kingdom

0. Abstract

This paper discusses general Human-Machine Interaction requirements for air traffic management (ATM) & avionics systems design. Human operators of these systems are not "average" users, and the operational conditions for these systems are carefully monitored and controlled. Having studied the characteristics of the users of ATM & avionics systems, we propose HMI improvements that increase performance and identify design trends that should be avoided, due to the peculiarities of the avionics operational environment.

1. Introduction

The objective of this paper[1] is to provide general HMI (Human-Machine Interaction) requirements for avionics system design; that is to say, a clear description of the general principles as well as the current trends and current research results relative to the HMI in the different fields of the avionics environment.

This paper is written from the viewpoint of the HMI researcher involved in the AFM (Air Flight Management) and ATC (Air Traffic Control) fields. Its intended audience are avionics systems designers and engineers, as well as HMI researchers who have an interest in applications of HF (human factors) theories in the avionics field. We will try to achieve two aims:

- identify some of the general characteristics ("profile") of the human operators (on board an aircraft or on the ground) of avionics systems. This is important as innovations or changes in the field of avionics will most probably have an effect on them. We will then use this profile of the operators to derive some general human machine interaction requirements and principles applicable in the different fields (areas of interest with common characteristics) of avionics.
- give an indication of the current HMI trends and research approaches relevant to each of the above mentioned fields of avionics.

It must be stressed here that defining the profile of the human operators of avionics systems is not enough to "drive" the HMI requirements for any avionics scenario; we need much more detailed information about the particular tasks of the humans involved in it. It is, however a valuable starting point, in the sense that it helps us decide which principles and HCI requirements are applicable or relevant to the avionics field.

2. A general profile of the operators in the ATM environment

In carrying out human factors or HMI research on a given population in a given environment, we must bear in mind that there are a number of factors that might influence the manner in which research results might be interpreted and generalised. We will briefly outline some of these factors.

2.1. Differences between ATM etc. personnel and other types of population

Broadly, there are two types of human operators (or actors) to be considered in the context of an avionics system. Those in the airborne part and those on the ground part of the system (control centre etc.). One fact that can be assumed for both categories of actors is they are not "average" users. They are well trained professionals, with well established skills. Furthermore, they are highly motivated users, in the sense that they will never be "casual users" or "lazy" in adapting to new technology. Their training makes it possible to use complex display formats

[1] A large part of the work described in this paper has been undertaken by the authors as part of the European Community AEGIS project, within the framework of the Community Research Programme, with a financial contribution from the European Commission. The following companies and Universities are involved in the AEGIS project: Aerospatiele, BNR Europe Ltd., Isdefe SA, National Technical University of Athens, Queen Mary and Westfield College (University of London), Sextant Avionique, Sofreavia and Syseca SA.

and symbology. Both pilots and controllers tend to be well educated, fit, carefully selected individuals, and irrespectivelly of their country of origin, they all speak the international language of aviation - English.

To make the peculiarities of the population of human operators in an ATM system (organisation) clearer, compare it with any other typical users' population, e.g. the car-driving population. The car-driving population spans a wide age and ability range; it is not subject to regular medical or proficiency tests; does not have to keep up with the changes in the automotive technology; includes cultural and sub-cultural groups that do not even speak the same language.

2.2. Operational conditions

There are tight timing requirements and safety requirements in the avionics organisational environment, which is very rigorously structured. There are, for example, protocols that control the communication activities between the human agents, ATCOs (Air Traffic Control Officers) and pilots. There is also a strict division of responsibilities among members of the crew or the ATCOs. Regular maintenance of equipment can be assumed. All devices and equipment can be assumed to be above the "average" standard of reliability and accuracy. The environment is also particular in the sense that it is controlled with regard to factors such as sound level and temperature, both for airborne and ground personnel. Lighting conditions can be carefully controlled.

To go back to the example of the previous sub-section, contrast these operational conditions with the ones applicable to the car-driving population. Vehicles can be operated in a variety of surfaces, with or without the windows down, with or without the car stereo on. Instrument condition is also dramatically different. We are all aware of drivers that simply ignore their wavering speedometers, or the oil pressure warning lights, for example. In practice, car drivers rely on information which is usually degraded in a degree that would make it unacceptable in the world of avionics.

2.3. Other unique features of the avionics environment

Furthermore, there are unique features of an aircraft operation, that set special HMI requirements. For example the dimensional space of an aircraft is 3-D with both translation and rotation. Instrumentation is of primary importance as opposed to, say, the car, where the instruments are of much less importance compared with the outside scene. The outside view is only one element of a pilot's scan pattern, which always includes the instrument panel - in particular in bad weather, where the pilot's attention focuses exclusively on the instruments. In a car, in contrast, the most important visual display is the outside scene, with the line markings, traffic lights, signs, other traffic etc.

3. HMI requirements and current trends

We shall now present the general HMI requirements and current HMI trends applicable to the various fields of an ATM system (organisation). These requirements and current trends will be grouped together with regard to their relevance to general HMI problem areas, e.g. presentation of information, degree of automation, intelligent interfaces etc.

3.1. Presentation of Information

Let us examine two of the main roles of human operators in the AFM field: pilot and controller. It is required by both these roles for the human to process a wide range of information presented in various forms; this information can be auditory (e.g. a speech synthesiser "speaks" a warning to the pilot), visual (e.g. a flashing red light or the iconic representations of two aircraft on a Air Traffic Control Officer's display moving towards each other), textual (e.g. text on a display) or tactile (e.g. the degree of "stiffness" of an old aircraft's steering column relaying some information about the speed/angle of turn combination).

New systems should be flexible as to the medium chosen for a particular type of information. Humans perceive information through different channels of perception, and there is a limit on the amount of perceptual load that can be handled by each of these channels. If the load on one of the channels approaches the upper limit, the system should be flexible enough to change the type of information. For example, assume that at a particular point in time an emergency situation arises (e. g. due to a failure in one of the engines of an aircraft) and that there are four dials which the pilot must monitor closely to cope with the emergency. If another piece of information arrives at the same time, e.g. a drop in oil pressure, the system should decide whether to present it as an indication in a fifth dial, which might overload the pilot's visual perception channel, or as e.g. an audio warning. The tasks of the pilot involve processing this information either serially or in parallel (in the sense of multi-threading or interleaving their activities). Serial processing should reflect the structure of the task, where parallel processing (when and where possible) increases pilot performance.

The operators will also be required to scan visual displays. The scan pattern reflects the operation's information needs. They change with experience, in the sense that the experienced operator will do many cross checks when time is available. The designer of the visual display should take into account what kind of information the operator needs during task performance in order to minimise the total distance travelled during a visual scan. Frequently needed information (e.g. altitude and speed, in the case of the pilot) should be placed centrally and close together. Priority should also be given to urgent information.

We mentioned before that parallel processing of information by the operator can increase performance. There are certain characteristics of information channels that seem to enhance or reduce the likelihood or probability that parallel processing will take place. Briefly, these are:

Ease of processing: to the extent that patterns of information consistently give rise to the same response, such information can be processed with very little cost to attention resources.

Proximity of display: neighbouring information has to be relevant to be processed in parallel.

Integration of information: different dimensions of the same object can be processed in parallel. Therefore it is preferable to integrate information about single objects rather than have large banks of displays one for each dimension of the object each requiring their own channel of information.

Peripheral Displays: it is now generally agreed that in conditions of stress the focus of attention is narrowed and changes in peripheral vision may be less well detected. The effect on performance of peripheral displays is still not exactly known. It is therefore suggested that they are do not fall into the category of those characteristics that enhance the possibility of parallel information processing.

3.2. Multi-modal interfaces.

As mentioned earlier, perception and processing of information in parallel can enhance the operators' performance. For the perception of information in parallel it is recommended that multiple modalities are used e.g. audio/visual. The use of multiple modalities of output and input is crucial for effective communications. In recent attempts (e.g. ODID simulation, ODID III, 1991) different media & modalities have been successfully used in an improved, fully automated ATC position.

Recently a method for the design of multimedia systems was proposed by Faraday (1993) which aims to support the design of multimedia to support users' tasks. The users' tasks are analysed and a TKS model (Johnson et al. 1988) is used to decompose each task into low level task actions. The desired information requirements for each action are established. Part of the design process is allocating media resources of the system to each of the actions. Design is performed in accord with design and cognitive heuristics.

No systematic means have been proposed so far for the assignment of media to information requirements within a task, although there have been experimental studies and some pragmatic guidelines suggested, e.g. Alty et. al. (1993). For example, spatial information and large quantity of numerical data are better presented graphically. Animated graphics can highlight important parts of the display, i.e. they function as attention grabbers. Animation is useful when the time dimension is important for the information presented. Recorded video is useful where realism is important.

Turning to audio displays, we can identify two main categories: speech displays and non-speech displays, where the information is coded. The latter is advantageous in the particular domain where operators are assumed to be highly trained. While the choice between visual and speech display is difficult and dependent upon the particular context, in general speech displays can be used for status information, warnings, issue of commands, feedback, etc. In general presentation of spatial information is best suited to the visual channel of perception.

Such guidelines are necessarily left vague, as the nature of the information many times does not allow a choice of modality for its presentation. The choice of modality (audio, visual, tactile etc.) and media (graphics, text, speech, non-speech) for a particular piece of information is particularly relevant to the engagement of perception channels of the user in the course of task performance.

When the visual channel is cluttered to the extend that any additional visual information might be lost, the audio channel is the most powerful alternative — especially in cases where eye fixation is required at a particular point for the performance of the task, or when vision is degraded (for example, darkness or glare may inhibit vision, especially when using a CRT display). It is easy to clutter the audio channel so care should be given to the allocation of signals to this channel. The properties of the technology used are extremely important to its effectiveness: intelligibility, speech rate, naturalness of the speech, pitch, synthesised vs. natural speech etc. Environmental factors, e.g. ambient noise are also extremely important.

The effectiveness of a particular display medium is dependent upon the particular task for which it is used, the operator to whom it is addressed and the environment. Results from research in this area tend to be task and problem specific and cannot be generalised.

In the domain of ATM, performance and safety are crucial considerations. Organisation and design of the display are not aiming to be intuitive and "user friendly" but rather to economise on the cognitive resources that the user has to allocate to the device. This is achieved by good design of symbols, layout that reflects the order of tasks to be performed and maximises the effectiveness of the users' perception, through parallel processing of the information presented to him/her.

3.3. Intelligent Displays

This relates to one of the fundamental requirements of HCI, that is that data must be displayed in such a way that it helps the user perform their task. Consider, for example, the prime goal of the overall ATC controllers' task, i.e. the safe and expeditious flow of air traffic through the airspace. To help the users in performing this task, the data must be displayed in a way that would help the controllers avoid conflicts by detecting them at an early stage. Information must be filtered so that control problems are identified and displayed in an unambiguous way.

This requirement has been partially implemented in ODID III (1991), where a part of the ATC display is occupied by the conflict detection window, which calculates the trajectory of each of the aircraft in the controller's sector and displays graphically pairs of aircraft that are in conflicting courses.

Another example of the move towards an “intelligent display” is the Command display. Broadly, we can distinguish between Status displays and Command displays. The status display informs the operator about the state of the system, while the command display assists the operator in controlling the system. Status displays require cognitive effort to translate into a decision. The objective of the command display is to make a computer do these calculations and provide the operator with information stating exactly what control inputs are required.

Another aspect of intelligent displays that might prove very helpful is prediction and anticipation. It has been noted that because of the “sluggishness” of aviation systems, if an operator responds to the current size of an error rather than to its rate of change of growth, the response may be too late. Actions must be based at any point on the estimated future error. Display technology must incorporate techniques to assist the operator, e.g. direct presentation of the rate of change of system state as well as its acceleration. It must be pointed out though that a straight forward “display quickening” technique, as a simplified version of this is known, has a number of drawbacks, and perhaps the major one is that the operator is not aware of the real current state of the system.

A better alternative is to provide information as to how to respond to the predicted state, while in fact presenting the current state of the user (predictive display). Predictive displays are designed to provide the operator with one or more symbols depicting the future state of the system output on the basis of certain assumptions made concerning the operators' future control activity. Predictive displays have been demonstrated unequivocally to be useful but their benefits are proportional to the accuracy of the prediction. The main drawback of predictive displays is that they add an extra dimension (that of time) to be displayed along with the status data.

Another useful property of the display (interface) is the passive monitoring of user attitude. For example the system could monitor the users' actions: is the operator talking to a colleague, is he/she looking at the screen? Answers to questions like these could be useful input to designing dialogues, for example when to alert the user, when not to interrupt, etc.

3.4. Degree of automation

There has been a lot of discussion on the degree of automation of safety-critical systems, and there are good arguments supporting both trends. Those in favour of fully automated systems argue that systems do not suffer from fatigue or eye strain; their reaction to a given set of stimuli is not affected by emotions; their reaction time is minimal. On the other hand, given that such systems are usually rule-driven, one must make sure that they "know" all the rules; that they can react to a new situation with a sensible action and not with an error message; that they are adaptable, in the sense that they enrich their rule domain as new situations occur.

Keeping the humans in the loop is a very realistic option especially considering the unpredictability of emergency situations, where human judgement is needed. It is appropriate to look into ways of allocating effort between human and machine in the most efficient way. There are certain issues that one can take into account.

Humans should not exceed a certain upper limit of workload; tasks that the humans enjoy doing should not be allocated to machines; tasks that keep the humans vigilant should not be allocated to machines. On the other hand, repetitive, routine tasks that are likely to appear tedious or uninteresting to humans, and therefore contribute to stress or situation of increased fatigue should be performed by machines. The main objective is to use the technology to support the persons' tasks. This requires us to understand the details of those tasks and the demands on the person. The machine should reduce rather than add to those demands, while increasing rather than decreasing the quality of and efficiency in which the task is performed. For too many systems increase the load on the human and decrease the quality of the task output.

One of the main dangers of automation is the temptation to automate those systems or parts of a system that are *easy to automate* rather than the ones that, from a human-factors perspective, *need to be automated.* There are a number of "rules" for deciding how to automate the system: flight deck automation can be introduced as a layered process (e.g. automation of control and automation of monitoring functions, Wiener 1988); automation can be classified into that which replaces human performance and that which merely assists the human operator by providing aids to resolve certain bottle-necks of human performance (Wickens 1990).

Function (or task) allocation is probably the most crucial question in this area. The general requirements from the automation strategy are that the overall performance is increased, that the vigilance of the operators remains high and that the strategy is reliable in crisis situations, where either system or human partner malfunctions. To this effect we must take into account the limitations of today's technology (e.g. the fact that avionics displays' technology might be inadequate to deal with proposed automated systems and may lead to a number of negative effects), as well as the effect on the operators (e.g. the false sense of security and the relaxation of their attention levels).

3.5. Human errors

There isn't a complete model of human action to explain to us the causes of erroneous action. Nevertheless such a theory is not necessary for increasing the robustness of the Human-machine interaction. For example, for recoverable actions an undo facility is necessary, for irrecoverable actions e.g. of a pilot in an airplane such actions should be anticipated at the design stage and barred by the designer.

Human errors may be attributed to two factors: either the interface does not cope with the psychological mechanisms of communication or there are discrepancies between the mental model the user forms of the device and the model of the user implicitly (and rarely explicitly) embodied in the device. There is always a user model embodied in the device: at the most basic level that the user can see the screen, can hear etc. At a more cognitive level, with respect to say the mathematical knowledge, the knowledge of the task domain, or even the intentions of the user at a particular point in the interaction.

With respect to the communication attributes of the interface, ergonomic guidelines can be used to cope with human perception restrictions. E.g. size and distance of displays, colours and contrasts, sound levels and noise, luminance of signals and background are basic considerations. Further the information conveyed by symbols has to be designed with similar care, see Wiener (1987). Guide-lines can be derived from consideration of the attention grabbing capability of system messages, or the limited capacity of working memory.

The second kind of failure in human machine communication results from discrepancies between the conceptual model of the interface and the mental model the user is applying. Two approaches can be suggested to get around this problem. That the task models of users are taken into account in the design of the actual interface. That the mental model that the user acquires of a particular device, i.e. the task model of conjoint task execution with the device is correct. Task analytic techniques can then be applied for the elicitation of this model. This line of thinking though obscures the fact that in cases of erroneous behaviour the "normal" or "correct" course of actions is not followed. The possible discrepancies are the subject of study of human error.

A systematic description of errors can tell us what can go wrong. Then the design should cover the possibility of correction, or the options that have to be eliminated in order to prevent the error ever occurring. Possible human errors can be classified in error modes, i.e. with respect to their manifestations. Error modes identified are:

- timing (too soon, too late)
- sequence (omission, jumps)
- type (incorrect type of action)
- force (too strong, too weak)
- duration (too long, too short)
- direction (too far, too short, wrong direction)
- object (wrong object)

These types of error refer to the actual user input, which can then be further examined along the modalities in which it occurs. In the case of sound for example force will be amplitude (loud, quiet) etc.

Human erroneous actions can then be further distinguished with respect to the consequences, and with respect to their causes. There is little theoretical coverage of the causes of error. A review of methodologies for root cause analysis of errors can be found in Gojazzi (1993). The following subsection discusses the case of cognitive errors based on mistakes of the human problem solving mechanism, which lead to the development of the expert critiquing technology.

Models of human operators' behaviour in dynamic situations have been constructed e.g. the COSIMO model in Cacciube, P., C., (1992), that can predict the human performance and its errors. Unfortunately while the process of constructing such models has shed light into human cognition it is not very clear how these models can feed into the design of man machine systems, as is pointed out in Decortis, F., (1993).

3.6. Human errors / Expert critiquing.

The effect of human error caused by misconception or by mistaken knowledge can be aggravated by a system not informing the user of the dangerous situation developing until it becomes too late. In many cases the reasoning process of the human, characterised by the knowledge utilised and the inference procedure can be flawed.

An expert critiquing system could identify such flaws and thus warn of a dangerous situation. As was mentioned before the human operator is an expert, so of particular interest to an ATM system (organisation) is the study, detection and correction of expert error. The study of expert error is a relatively new field, as literature in psychology and HCI is concerned traditionally with novice error.

Expert critiquing systems are defined as facilities which support systems for complex tasks. The system behaves as an expert critique, possibly collaborating with the user to solve a problem and advance the task with added quality.

This involves the system monitoring what the user is undertaking and helping the user by pointing out "mistakes", "errors", and/or "misconceptions" (where there is sufficient evidence that these exist); for instance, when a controller assumes that an aircraft is at a particular point in the sector or at a particular altitude, and the pilot has been told to fly at a particular level and yet she/he is off course. Here, there will be a discrepancy between where the aircraft actually is and where the aircraft is assumed to be. It would be possible for the system to point out the nature of the error and propose (if necessary) carry out the necessary action. Not only must the system have a model of the user's task but also must monitor the dialogue between air and ground and be able to identify what is wrong, as a mismatch between what should be happening and what is actually happening.

Silverman et. al. (1992) identify a set of principles for effective critique styles:

- Critics are most useful in a pre-existing automated environment for users attempting semi-structured tasks.
- Critics should be deployed with considerations for timing (before, during or after the task), process (incremental or batch), mode (active i.e. self triggered or passive), knowledge (shallow, deep models of knowledge), algorithm (heuristic, formal etc.) and interface(media).
- Critics must attempt to cause the individual to notice and use more cues (because human attention is selective).
- Critics should help the individual ingrate his/her own intuition with correct cues in a manner consistent with formal reasoning procedure.

Expert critiquing technology is not mature. Critiquing technology of today is characterised by post error explanation which is generally not acceptable. Critique is presented in textual format. Studies in safety critical and real time domains have not been presented. Further there is limited work on modelling the expert error and current critiques do not have a user modelling component, and are not adaptive. As a concept expert critiquing systems are promising but still futuristic in the context of air traffic control.

3.7. Support of co-operation between users

There exist many examples of user co-operation in the ATC/ATFM field. They include planning controller/executive controller co-operation, pilot/co-pilot co-operation, pilot/flight engineer etc. Traditionally this is done either verbally (pilot talking to co-pilot or flight engineer) or by exchange of hand written data (e.g. on flight strips).

In modern systems though, there are tendencies towards supporting automated co-operation between users. For example, in the ODID III simulation, the planning controller communicates an AOC (Assume Of Control) message to the executive controller. This message appears on the executive controllers screen who can accept it as is or request changes. Thus, verbal communication (which could be problematic in noisy conditions) or hand-written communications (which could be misinterpreted) are avoided. Communication facilities such as voice, video, text must be integrated to provide asynchronous and synchronous communication to support human-human collaboration on co-operative tasks.

3.8. Input devices

There exists an immense variety of input devices. As yet there is no systematic way of prescribing the appropriate input device for a particular task and task context. There have been several attempts to produce a taxonomy of input devices, for graphical input by Buxton (1990), and more generally man machine interfaces by Card (1990). The latter is particularly interesting as it provides a general model of input devices, i.e. a model of the choices possible for the design of an input device. Designed devices can then be judged in terms of their expressiveness and their effectiveness.

There is a multitude of empirical studies measuring the performance of these devices. Some reports on the use of novel input techniques are listed below. The presentation of several studies on the combination of some of these input devices into multiple modality input technique can be found in Blattner and Dannenberg (1992):

- simultaneous two handed input. Most current workstations are considering single stream input and are not capable of tracking two devices, as would be required by two handed input, which nevertheless seems a surprisingly natural technique for interaction, Buxton (1986).

- stereo display in conjunction with three dimensional manipulation. Three dimensional input devices include the data glove, or the bat i.e. the three dimensional mouse Slater, M., and Davison, A., (1991). Research in three dimensional input has also considered the recognition of gestures, e.g. Wolf, C., G., (1992) for an empirical comparison of gestural, keyboard and mouse interfaces.

- speech input is already a popular input technique. Current research is focusing on the co-ordination of speech with manual input. In Blattner and Dannenberg (1992) the combination of speech with gestural input is discussed. Also in Schmandt (1982) the combination of pointing devices and speech input is discussed.

- eye input technology. Eye trackers are a futuristic input device proposed. They are not as reliable as one would desire for the operation of real time systems. Further careful consideration is required for the design of interaction techniques that will utilise these devices. An evaluation of eye tracker input devices is presented in Ware (1987).

When it comes to reported results for the particular domain of ATCo and of on board systems, our main source of information is the ODID III simulation where a new ATC workstation was evaluated. One of the characteristics of this workstation was that the way of interaction with the system, in terms of input devices, was unique: it was all done using a two-button mouse. In theory, this should improve the quality of the interaction, as the users do not have to switch input devices to input different types of information.

During the evaluation of this workstation, the majority of the controllers felt that one of the drawbacks of the design was that it only provided one way of inputting information! They all preferred to have a keyboard in addition to the mouse, and have the freedom to use whichever input device seems convenient at a given time. We can therefore suggest that the input device must be flexible enough to comply with the users' requirement.

In the case of the aircraft cockpit instrumentation, we can note the following:

- voice control. The quality of the speech input technology is critical to the viability of this input technique, as errors arise not from operator error but from the unreliability and slowness of the speech recognition technology. Keyboard entry is faster but nevertheless users in stressed tasks prefer voice entry. (Christ and Malkin 1985).

- keyboard data entry. Advantageous for input large amounts of data, especially for alphanumeric sequences are more rapid and accurate. Less accurate than traditional controls like switches, toggles, thumb wheels etc.

- touch control has been suggested to provide maximum hand-eye co-ordination, control -display compatibility, Beringer (1983).

Summarising it is obvious that no single modality is sufficient. However in combination with other modalities which are either better suited for certain kinds of input or compensate for certain deficiencies of the technology e.g. of speech or gesture recognition, interaction is improved.

3.9. Multiple task performance and workload

There are two main issues to be taken into account here: the variables that affect the ability to perform multiple task concurrently; and the parameters that affect workload.

Briefly, most models of task time-sharing take into account the task difficulty, sometimes expressed in the framework of a performance-resource function; the task demand for processing structures; the processing modalities, e.g. the fact that it is easier to time-share an auditory and a visual task than two auditory or two visual tasks; task similarity and integration; task queuing and scheduling etc.

On the other hand, good task performance is not satisfactory enough if it imposes a massive mental workload on the human operator. There are a number of ways for controlling or assessing workload. One can make sure that in designing a control position or a cockpit, all the secondary tasks are not obtrusive, i.e. they can be accomplished with minimum disruption of the primary tasks. We can use empirical methods to evaluate workload (one of the oldest and best validated subjective measure of pilot workload is the Cooper-Harper rating scale of aircraft handling qualities, Cooper and Harper 1969).

4. Conclusion

We have presented here a brief summary of the general HMI requirements and current trends applicable in the different fields of an ATM system (organisation). To this effect, we have attempted to present some general characteristics of the human operators involved in the control loop where the innovations of the avionics industry will most probably have an effect. This "profile" of the operators influenced the choice of the general human machine interaction requirements applicable in the different fields (areas of interest with common characteristics) of an ATM system (organisation). We have also given an outline of the current HMI trends and research approaches relevant to the different fields of the system.

5. References

Alty, J L, Bergan, M & Craufurd, P., *Experiments Using Multimedia Interfaces in Process Control: Some Initial Results,* Computer Graphics, vol., 17., No., 3, pp.205-218, 1993.

Beringer, D B, (1983). The pilot computer direct access interface: touch panel revisited. Human Factors 27(4).

Blattner, M M, and Dannenberg, R B, Multimedia Interface Design, ACM Press, 1992.

Buxton, W A S, (1990). A Three State Model of Graphical Input, in D.Diaper et al.(eds), Interact'90 Conference Proceedings, Elsevier Science Publishers B.V. (North-Holland), IFIP, pp 449-456.

Buxton, W, Myers, B A., A study in two handed input, Proceedings of the ACM CHI'86 Human Factors in Computing Systems Conference, 321-326.

Card, S, Mackinlay, J D, Robertson, G G. (1991). A Morphological Analysis of the Design space of Input Devices, ACM Transactions on Information systems, Vol.9, No.2, pp. 99-122.

Cojazzi, G, (1993). Root Cause Analysis Methodologies. Selection Criteria and Preliminary Evaluation. ISPRA Technical Note No.I.93.93 ISEI/IE 2442./93.

Cooper, G E and Harper, R P (1969): The use of pilot ratings in the evaluation of aircraft handling qualities. NASA Ames technical report NASA -TN-D-5153

Faraday, P Sutcliffe, A. (1993). A Method for Multimedia Interface Design, in Alty J L Diaper, D Guest, S People and computers VIII, pp. 173-190.

Johnson et al. (1988) P. Johnson, H. Johnson, R. Waddington, A. Shouls. Task Related Knowledge Structures: Analysis, Modelling and Application. In: D.M. Jones and R. Winder (eds.), *People and Computers: From Research to Implementation, HCI '88*, Cambridge University Press, pp. 137-155.

ODID III (1991): Eurocontrol EEC Report No. 242, "ODID III Real Time Simulation", Brussells 1991.

Silverman, B G, Mezher, T M (1992): Expert Critics in Engineering Design: Lessons Learned and Research Needs, AI Magazine, Vol.13, No. 2, Spring 1992, pp.45-62.

Slater, M Davison, A (1991). Liberation from flatland: 3D Interaction Based on the Desktop Bat, Eurographics'1991 conference proceedings, North Holland, 1991.

Ware, C & Mikaelian, H T (1987), An evaluation of an eye tracker as a device for computer input, Proc. ACM CHI+GI'87, Human Factors in Computing Systems Conference, 182-188.

Wickens, C D (1990): Engineering Psychology and Human Performance, Proc. of 32nd meeting of the HF society, Santa Monica, CA

Wiener, E L (1988): Human Factors in Flight, Academic Press Inc San Diego, CA.

Wiener, E L: Fallible Humans and Vulnerable Systems: Lessons Learned from Aviation, in Wise, J A, Deboms, A (eds) NATO ASI series, Vol. F32, Information Systems: Failure Analysis, Springer Verlag, Berlin-Heiselberg 1987.

Wolf, C G (1992). A comparative study of gestural, keyboard, and mouse interfaces, Behaviour and information Technology, Vol. 11 No 1, January 1992, pp.43-73.

AIAA-94-3408-CP

ADA, AUTOCODING, AND OBJECT-BASED SOFTWARE DESIGN - AN APPLICATION TO REAL-TIME FLIGHT CONTROLS SIMULATION

Ben Green*
Lockheed Aeronautical Systems Company - LASC
Marietta, Georgia

Abstract

This paper will discuss this authors objective and subjective impressions and conclusions of Ada as it is applied to the design of a real-time pilot-in-the-loop flight control simulation for a large military transport aircraft. The tools developed and design methologies used are different from anything in the past due to the Ada language and the advanced host computer system architecture. The object-oriented design of the simulation software will be discussed as well as the new simulation utilities developed to conform to the style and form of the Ada object-based design paradigm. The development of the actual software code involved the use of an autocoding tool used early in the development process. This will be discussed and conclusions are drawn regarding the value of autocoding and how it can be improved.

Background

Although Ada has been in service for many years it is just beginning to be applied to simulation of aircraft, avionics, and flight control systems. There are few tools and methods available for Ada because most past experience is coding with FORTRAN. Because of the capabilities of Ada over FORTRAN, it would would be a waste to code in Ada using outdated methods and tools even if they could be converted one-to-one.

As late as 1988, real-time aircraft simulation code , which included the airframe as well as propulsion, flight controls, and avionics, was still using assembler language for some applications The host computer was a small, single processor, microprocessor computer system used to simulate a large Class III military transport aircraft primarily for engineering or system development rather than pilot training. As can be expected, the choice of the update rate or computation interval was limited by the speed of the processor and this restricted the degree of fidelity that could be incorporated into the flight control system models. The code was not readable by anyone other than the programmer and was not transportable.

In the mid 80's LASC simulation capability was upgraded to include a motion base and an electronic visual system . At the same time the host simulation computer was upgraded to a specialized simulator minicomputer . The language of choice or necessity at that time was FORTRAN. This was a giant leap over assembler language because it improved readability and transportability, but it fell short in tenns of modern programming capabilities and methods. It does not offer many of the features of Ada or C++ that make it suitable for object-oriented design or long-term cost-effective software development and maintenance . It had far to go before achieving even modest goals proclaimed by prophets of Ada and object-oriented design.

Several years ago , LASC simulation capability was again upgraded to a multiple board, multiprocessing VME bus chassis system for real-time simulation. Processing resources could then be tailored to the application and Ada would be used for the high level language. Of the two languages in consideration for object-oriented design , C++ and Ada , Ada appears to be the closest to FORTRAN in syntax structure and appearance while offering far greater potential . In my opinion and the opinion of others C and C++ still have a "brain damaged syntax"[†]that hampers readability. Since the government is mandating Ada for government programs, Ada is the logical choice.

While this paper will not deal with a rigorous methodology, it will apply concepts for designing elements that fit into an object-oriented scheme for simulation of flight control systems. Concepts such as abstraction, hiding,

*Flight Control Specialist Engineer
Member AIAA

[†]quoted from Reference 1 , Preface, paragraph 2, line 4

module coupling , generics, state data security, etc. will be used to develop these components. There will be two types of objects shown. The first are objects that are specific to the airplane and cannot be reused although the concepts of the structure can be used over and over. The second is the reusable utility functions and procedures which can be reused: integrators , filters , math functions , etc.

Object-Oriented Programming

Even my short amount of experience with large simulation software systems teaches me that there is "no silver Bullet"[‡] . In other words . Ada, object-oriented design, and autocoding are not going to solve all the problems and make straight the road of developing large software systems. They will not result in any rapid drop in software costs partly due to the fact that software itself is complex, it requires large numbers of people working together who cannot always work together, and because as our capabilities improve so does the size and complexity of the software system.[2]

Poor programming practices can still be used with Ada and object-oriented programming (OOP) does not guarantee perfection; however, they can be of great benefit to a reusable, transportable, maintainable simulation that can stand the test of time.

The discussion to follow is not written by an expert in OOP or object-oriented design (OOD) proclaiming the virtues of OOP while at the same time writing the paper in such a manner that the diet is too rich for the common peasants.. Very often the technical discussion is too theoretical, steeped in terminology and verbose dialogue and with a level of abstraction that prevents seeing it through to real world applications. . It is time for those on high to roll some stone tablets down to the sinners. Do not get the wrong impression. Theory is the underlying basis for all engineering, but as they say in the aircraft business, " you have to put rubber on the ramp". I will write this paper from a layman's perspective of which I am one of the congregation.

For many years I heard of OOP but was unable to find a clear explanation of OOP or how it could be used to improve software design. About the same time I started to use Ada in an OOD environment I found a simple yet complete (at least for my purpose) description of OOP which answered most of my questions. This book[1] is a C++ programming book geared toward professional programmers with experience in C. It is described as an introduction to OOP with extensions to C and C++. I will draw heavily from this book in this description of what OOP means to me and its relationship to the OOD flight control system simulation application.

o Data Abstraction

"Back at the dawn of history - about three years ago - object-oriented programming was called *data abstraction* and data abstraction is still central to the loose collection of programming techniques banded under the OOP flag The main idea of data abstraction is to isolate the data itself (and the mechanics of manipulating the data) from an application program."[§]

o Objects and Methods

"An object (a declared instance of a particular type - you're used to calling objects "variables") is manipulated entirely by means of subroutines called *methods* . A method is no different from any other subroutine except that it must be passed a pointer to the object being manipulated."[**]

o Objects and Messages

"The concept of data abstraction has now transmogrified into object-oriented programming. An object-oriented design looks at a program as a group of black boxes" ... The black boxes communicate by means of messages that travel along communication paths."[††]

o Classes and Objects

"A class has two components. The first is a data structure... The data component of a class collectively represent the object state, and is sometimes called the state data. The second component of a class is a group of subroutines called methods that manipulate the data component of the individual objects (the declared instance of the class). You communicate with an object by sending it a message made up of some sort of method selector and some optional data. That is, the message tells the object to apply one of its methods to itself, perhaps using additional data (that can be a part of the message) to do the work."[‡‡]

o Implementing Messages and Objects

" A message has three parts :

1. A *method selector* that tells the receiving object what to do - which method function to invoke.

[‡] quoted from Reference 2 , paragraph 3 , first sensence

[§] quoted from Reference 1 , page 2 , paragraph 1

[**] quoted from Reference 1 , page 2 , paragraph 2

[††] quoted from Reference 1 , page 3 , paragraph 1

[‡‡] quoted from Reference 1 , section 1.4 , paragraph 1,2,3 , page 5

2. ... - a process that receives a physical message.
... a receiver can be a physical data structure -

3. Optional data used by the method to do its task."[§§]

o Object Access
" An object should provide external access to its data"[***]

The preceding section serves to introduce the OOP jargon and concepts of the OOP paradigm. These concepts will be demonstrated in the object-oriented design (OOD) for a real-time aircraft simulation.[1]

Object-oriented Design for Real-Time Aircraft Simulation

The simulation described is abbreviated in scope for simplicity without loss of content. The actual simulation includes a typical avionics suite for a modern military or commercial transport aircraft. I will show only the basic airframe and flight controls. Figure 1 shows a high level diagram of the simulation software structure.

The SCHEDULER can be considered to be the main body of the applications program which is responsible for sequencing the subsystem in real-time. It also connects with the user interface to control the simulation modes such as initialize, trim, run, freeze, etc. The simulation is decomposed into subsystems which are sometimes referred to a objects. These are simulated at either a fast rate of 100 hertz or a slow rate of 10 hertz.

The SCHEDULER has three procedures, INITIALIZE, CONNECT, DISCONNECT, and two tasks 100_HZ_TASK and 10_HZ_TASK. The OOP paradigm states that the application program communicates with the objects (subsystems) using messages. This works well with programs that are asynchronous but this is a time synchronous system. Instead, the method selector is the control thread enabled by the _INTERFACE_PKG's This allows the method (procedure call) to proceed through one step. The input and output data to and from the object is passed through messages. The message utility will be discussed later.

In this example only the FCS (flight control system) is

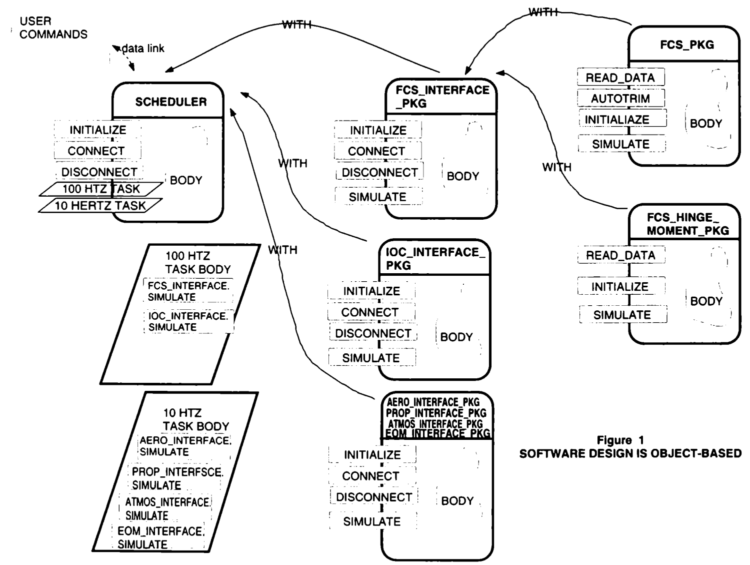


Figure 1
SOFTWARE DESIGN IS OBJECT-BASED

[§§] quoted from Reference 1 , section 1.5 , page 6 , paragraph 1

[***] quoted from Reference 1 , section 1.3 , page 3 , paragraph 1

shown in detail. The other subsystems shown are as follows.

FCS_IOC - interface between the FCS and the hardware control loading device.

AERO - aerodynamics
PROP - propulsion
ATMOS - atmospheric data
EOM - equations of motion

Each subsystem has an object which contains a declared instance of the subsystem or object state type. With the *methods* or procedures to manipulate the object data it forms a *class* . The *class* is conveniently enclosed within an Ada package such as shown in Figure 1 FCS_PKG. *Optional data* and *return data* is through the procedure argument of each *message*. Each package also has a corresponding _INTERFACE_PKG as in FCS_INTERFACE_PKG and the interface package is the mechanism through which the message passing , both *method selector and data,* is implemented.

There are basically four *methods* that are selected by the SCHEDULER in FCS_INTERFACE that manipulate the object data for the subsystem class in FCS_PKG.

INITIALIZE - initializes the object prior to simulation

CONNECT - connects the object to the message utility prior to simulation

DISCONNECT - disconnect the object from the message utility when simulation session ends

SIMULATE - simulate the object for one time step and update the object data (state)

In the SIMULATE *method* all the object data from all other objects that couple to the FCS as input data are retrieved from the message utility before FCS_PKG.SIMULATE is called. Once the state is updated the FCS state data is output to the message utility.

The FCS_INTERFACE_PKG directly manipulates the FCS_PKG methods, but even at this level it can only read the state data and manipulate it through predetermined methods. The control flow or threads from the SCHEDULER to the FCS_INTERFACE are direct for INITIALIZE, CONNECT, and DISCONNECT. Simulate is referenced from the 100 hertz task in the case of the FCS.

One aspect that is not revealed in Figure 2 is the multiprocessing architecture of the host computer system. This is shown in Figure 2 along with the message passing scheme. For the sake of this example each subsystem object is executed on a separate board . All the boards are connected by a VME bus. The message passing between the object is over the VME bus physically, but made transparent to the user by the abstraction of the message passing utility.

The SCHEDULER and the MESSAGE UTILITY runs on board #1. The SCHEDULER controls execution of the methods for each object and the objects communicate through the message passing utility with data. Both the data and control are over the VME bus which is transparent to both the subsystem object and the simulation application. The low level details are accounted for by the development system environment which are beyond the scope of this paper; however, this software architecture would run on a uniprocessor host given sufficient throughput capability.

The design of the FCS subsystem object is shown in Figure 3. The FCS_PKG is the package that forms a class by enclosing the object data and defining the methods (procedures) that act on that data. Since this is a single airframe simulation there is only one instance of the FCS object. The READ_DATA *method* reads the parametric data from a file for the FCS_PKG. AUTOTRIM is a procedure which invokes a subset of the flight control system equations which establish non-dynamic relationships between the pilot controls (column, wheel, pedals) and the control surfaces (elevator, aileron, rudder) so that the airframe trim subsystem , AUTO_TRIM (not shown), can trim the aircraft without possessing the details of the FCS. AUTO_TRIM is performed prior to simulation and trims the airplane in a a static sense (zero accelerations and rates) and should not be confused with any manual or automatic trim functions while in flight. INITIALIZE initializes the object data or state prior to executing or simulating the airframe dynamically in real-time. SIMULATE simulates the FCS by integrating and updating the model or object when called every 0.01 second time step.

The FCS is divided into four parts which are coded as packages: FCS_ELEVATOR_PKG , FCS_AILERON_PKG, FCS_AILERON_PKG, FCS_FLAP_PKG. Except for the flap each has four procedures in the specification that are one-to-one with the FCS_PKG methods and contain the details of each method for each individual axis. There are separate procedures called within the body of each which contain the details of the methods. READ_DATA, AUTO_TRIM, and INITIALIZE are one-

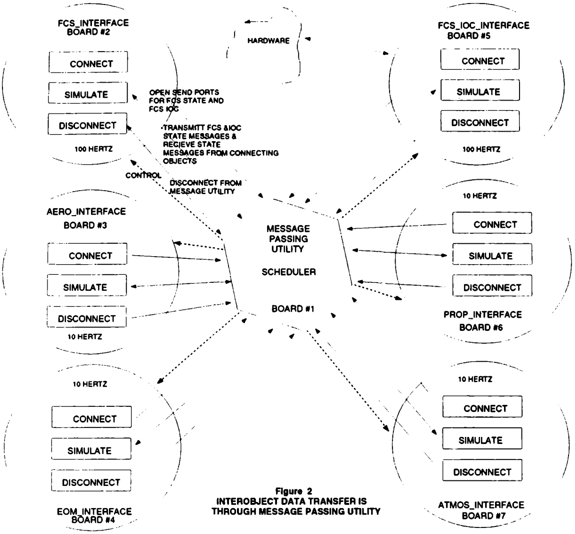


Figure 2
INTEROBJECT DATA TRANSFER IS THROUGH MESSAGE PASSING UTILITY

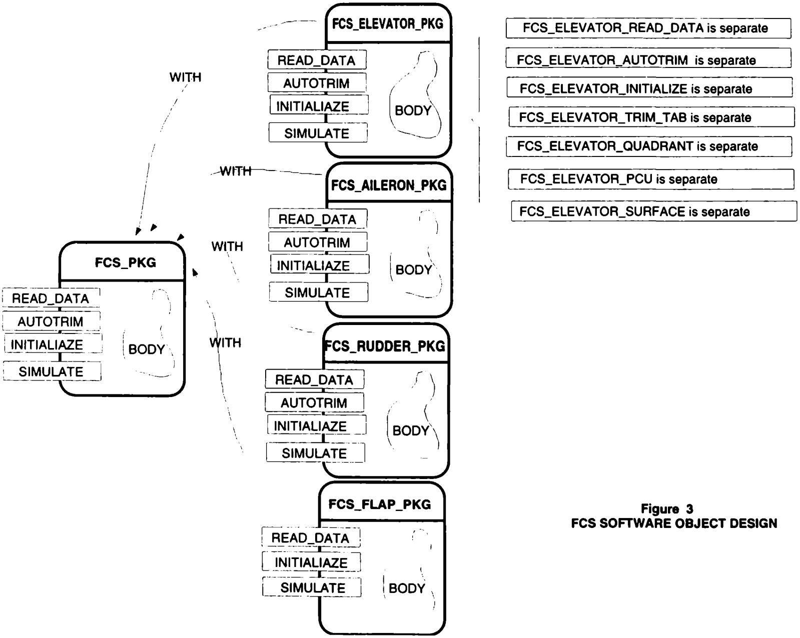


Figure 3
FCS SOFTWARE OBJECT DESIGN

to-one while TRIM_TAB, QUADRANT, PCU, and SURFACE define the simulation model for components of each axis and are called from SIMULATE (shown here for the elevator or pitch axis only).

There is an excessive nesting of procedure calls before getting to implementation details;however, they are straight forward and consistent. They are necessary to allow a step by step progression of the design and coding to proceed top-down and bottom-up. The development of the FCS_PKG can proceed without being influenced by the message passing protocol or the other objects. The FCS_INTERFACE_PKG controls the interface to the FCS_PKG and isolates the SCHEDULER from the details of the FCS_INTERFACE_PKG. The individual FCS subsystem packages and separate procedures are needed to reduce the actual code to a readable and manageable level.

Figure 4 shows the details of the FCS_PKG. FCS_PKG "withs" in FCS_TYPES which is also visible to the other subsystem objects (the type but not the actual data). A STATE_TYPE is defined for the FCS_PKG state data. This state type is the output of the transformed internal state of the FCS object that is needed by the other objects. In this case the AERO_PKG will need surface position, etc. FCS_TYPES is also "withed" in to the other interfaces such as AERO_INTERFACE so that the FCS_STATE can be received from the MESSAGE UTILITY by a pointer to that data message. Also defined in FCS_PKG are input and output types such as SIMULATE_INPUT_TYPE which allows the input data to the simulate method to be packaged together making the procedure argument less cumbersome. Whenever possible, data is packaged together in logical groups with record types which streamlines the transfer of data.

Figure 5. shows the block structure of the FCS_ELEVATOR_PKG body. The parametric data is declared so that it is visible to all procedures that are separate . Since the parametric data is read from a data file prior to simulation and remains constant there is little reason to pass it through procedure arguments. Next, the interprocedural variables are declared for all the separate procedure arguments. Next the body of the four procedures in the specification are defined. The separate procedures are defined before the body of the procedure they are called in. Although not

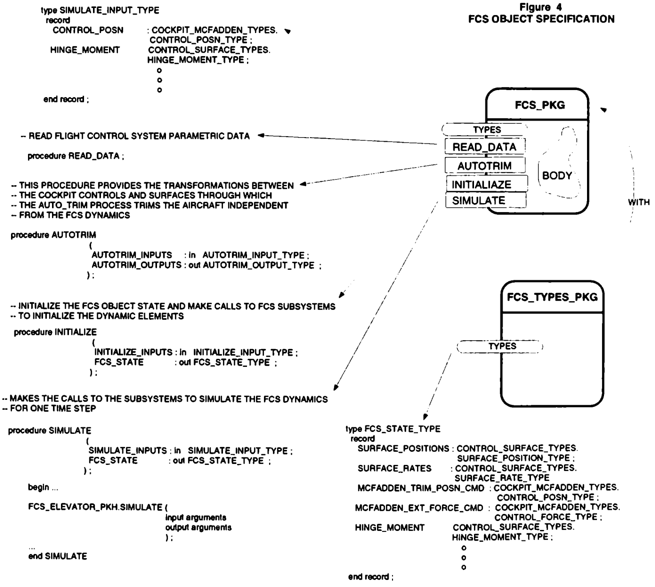


Figure 4
FCS OBJECT SPECIFICATION

```
type SIMULATE_INPUT_TYPE
 record
   CONTROL_POSN        : COCKPIT_MCFADDEN_TYPES.
                         CONTROL_POSN_TYPE ;
  HINGE_MOMENT          CONTROL_SURFACE_TYPES.
                        HINGE_MOMENT_TYPE ;
                          o
                          o
                          o
 end record ;

 -- READ FLIGHT CONTROL SYSTEM PARAMETRIC DATA

   procedure READ_DATA ;

-- THIS PROCEDURE PROVIDES THE TRANSFORMATIONS BETWEEN
-- THE COCKPIT CONTROLS AND SURFACES THROUGH WHICH
-- THE AUTO_TRIM PROCESS TRIMS THE AIRCRAFT INDEPENDENT
-- FROM THE FCS DYNAMICS

procedure AUTOTRIM
              (
               AUTOTRIM_INPUTS   : in  AUTOTRIM_INPUT_TYPE ;
               AUTOTRIM_OUTPUTS  : out AUTOTRIM_OUTPUT_TYPE ;
              ) ;

-- INITIALIZE THE FCS OBJECT STATE AND MAKE CALLS TO FCS SUBSYSTEMS
-- TO INITIALIZE THE DYNAMIC ELEMENTS

 procedure INITIALIZE
              (
               INITIALIZE_INPUTS : in  INITIALIZE_INPUT_TYPE ;
               FCS_STATE         : out FCS_STATE_TYPE ;
              ) ;

-- MAKES THE CALLS TO THE SUBSYSTEMS TO SIMULATE THE FCS DYNAMICS
-- FOR ONE TIME STEP

procedure SIMULATE
              (
               SIMULATE_INPUTS : in  SIMULATE_INPUT_TYPE ;
               FCS_STATE       : out FCS_STATE_TYPE ;
              ) ;

    begin ...

    FCS_ELEVATOR_PKH.SIMULATE (
                             input arguments
                             output arguments
                             ) ;
    ...
    end SIMULATE
```

```
type FCS_STATE_TYPE
 record
  SURFACE_POSITIONS : CONTROL_SURFACE_TYPES.
                      SURFACE_POSITION_TYPE ;
  SURFACE_RATES     : CONTROL_SURFACE_TYPES.
                      SURFACE_RATE_TYPE
  MCFADDEN_TRIM_POSN_CMD : COCKPIT_MCFADDEN_TYPES.
                           CONTROL_POSN_TYPE ;
  MCFADDEN_EXT_FORCE_CMD : COCKPIT_MCFADDEN_TYPES.
                           CONTROL_FORCE_TYPE ;
  HINGE_MOMENT        CONTROL_SURFACE_TYPES.
                      HINGE_MOMENT_TYPE ;
                        o
                        o
                        o
 end record ;
```

shown explicitly, each separate procedure has argument lists not shown due to space limitations. The procedure (*method*) SIMULATE is at the bottom of the package body and calls each of the four components of the simulation. The call to the ELEVATOR_TRIM_TAB is shown explicitly.

```
 package body FCS_ELEVATOR_PKG is

 --  parametric data declarations within scope of FCS_ELEVATOR
 -- data Is read in through FCS_ELEVATOR_READ_DATA
      XXXX
 -- interprocedural variable declarations  and types
     XXXX
procedure FCS_ELEVATOR_READ_DATA is separate ;
procedure  READ_DATA   is



 procedure FCS_ELEVATOR_AUTOTRIM( ) is separate ;
 procedure  AUTOTRIM (
                      AUTOTRIM_INPUT :    in      AUTOTRIM_INPUT_TYPE ;
                      AUTOTRIM_OUTPUT :  out   AUTOTRIM_OUTPUT_TYPE
                      )  is

 procedure FCS_ELEVATOR_INITIALIZE( ) is separate ;
 procedure  INITIALIZE (
                      INITIALIZE_INPUT :    in     INITIALIZE_INPUT_TYPE ;
                      INITIALIZE_OUTPUT :  out   INITIALIZE_OUTPUT_TYPE
                      )  is


 ; procedure FCS_ELEVATOR_TRIM_TAB ;
                      (
                      TRIM_TAB_INPUT :    in     TRIM_TAB_INPUT_TYPE ;
                      TRIM_TAB_OUTPUT :  out   TRIM_TAB_OUTPUT_TYPE
                      )  is separate ;

 procedure FCS_ELEVATOR_QUADRANT( ) is separate ;
 procedure FCS_ELEVATOR_PCU( )is separate
 procedure FCS_ELEVATOR_SURFACE( ) is separate ;;

  procedure  SIMULATE (
                       SIMULATE_INPUT :    in     SIMULATE_INPUT_TYPE ;
                       SIMULATE_OUTPUT :  out   SIMULATE_OUTPUT_TYPE
                       )  is

:  FCS_ELEVATOR_TRIM_TAB ;
                     (
                     TRIM_TAB_INPUT :    in     TRIM_TAB_INPUT_TYPE ;
                     TRIM_TAB_OUTPUT :  out   TRIM_TAB_OUTPUT_TYPE
                     )  ;

  FCS_ELEVATOR_QUADRANT(  );
  FCS_ELEVATOR_PCU( );
  FCS_ELEVATOR_SURFACE( ):

  end FCS_ELEVATOR_PKG ;
```

Figure 5
TYPICAL FCS SUBSYSTEM PACKAGE

Object-Oriented Utilities

I now come to one of the last features of the this OOD simulation: the dynamic utilities. In this case I have chosen an Euler integrator from one of many elements. It encompasses the concepts of the OOP paradigm.

(1) abstraction of low level implementation details

(2) protection of state data from external access

(3) forms a *class* consisting of an object type and methods to manipulate the object.

(4) communication with the object uses a method selector and input data list.

The integrator specification is shown in Figure 6 at the top of the figure. The variable type is abstracted to a unitless type (actually a floating point decimal number) for the integral and integrand. The time step is in units of seconds, and an integrator mode type defines the method selector. The actual integration is accomplished using a function call. The package for this utility is generic and an instance is created for each integration within a dynamic model. All of the state data , including constants are automatically accounted for each instance so the programmer is freed from state data bookkeeping. This becomes more important as the complexity of a utility function grows.

```
 -- - sample of a generic dynamic modelling element

 generic
 package EULER_INTEGRATOR is

  type INTEGRATOR_MODE_TYPE is
    (
    INITIALIZE_INTEGRAL_EQUAL_TO_THE INTEGRAND ,
   INTEGRATE_ONE_TIME_STEP_AND_RETURN_PRESENT_STATE ,
   INTEGRATE_ONE_TIME_STEP_AND _RETURN_NEXT_STATE ,
  FREEZE_INTEGRATION_AND_RETURN_PRESENT_STATE ,
  FREEZE_INTEGRATION_AND_RETURN_NEXT_STATE
   );

 subtype INTEGRAL_TYPE is UNIVERSAL_TYPES.UNITLESS ;

 procedure INITIALIZE
            (
            INITIAL_INTEGRAL_STATE : in  UNIVERSAL_TYPES.UNITLESS ;
            TIME_STEP                        : in  UNIVERSAL_TYPES.UNITLESS ;
             );

 function INTEGRAL_OF
           (
           INTEGRAND : in UNIVERSAL_TYPES.UNITLESS ;
           MODE          :  in UNIVERSAL_TYPES.UNITLESS
           )  return INTEGRAL_TYPE ;

 end   EULER_INTEGRATOR ;
--- instantiation of a dynamic modelling element

   package QUAD_POSN_INTEG is new  MODELLING_UTIL.EULER_INTEGRATOR ;

--  initialization of the dynamic element

  QUAD_POSN_INTEG.INITIALIZE
          (
          INITIAL_INTEGRAL_STATE => ELEVATOR_POSN / Gqs ;
          LOWER_LIMIT                    => QUADRANT_LOWER_POSN_LIMIT ,
          UPPER_LIMIT                    => QUADRANT_UPPER_POSN_LIMIT ,
          TIME_STEP                       => T_STEP
         );

 -- simulation of  a dynamic element

QUAD_POSN := QUAD_POSN_INTEG.INTEGRAL_OF
               (
               INTEGRAND =>  QUAD_RATE ,
               MODE          => QUAD_POSN_INTEG.
                                  INTEGRATE_ONE_TIME_STEP_AND_
                                  RETURN_PRESENT_STATE
```

Figure 5   TYPICAL FCS SUBSYSTEM PACKAGE

The procedure or *method* INITIALIZE is used to download the time step and initial state prior to simulation. Once the simulation is is running the integrator function INTEGRAL_OF is used to in one of five ways according to the method selector MODE. The INTEGRAND is the variable being integrated and is input to the function. One of the methods is to reinitialize the state equal to the INTEGRAND. The other methods either integrate one time step or freeze the integration while returning either the present state or the next state.

Autocoding With Ada

Before starting this section I want to say that the AutoCode[†††] tool which is a part of MATRIXx[‡‡‡]and was used to automatically generate Ada code for this paper is in my opinion the best commercially available autocoding tool at this time. I was not able to fully exploit the full capabilities of the tool in the time available so the process was not refined or improved according to all the options available. I am stating this to make it perfectly clear that any negative or constructive criticism of autocoding in this paper is not an indictment of this tool but of all autocoding tools in general.

The model for the elevator, aileron and rudder FCS were developed and tested from a block diagram specification using MATRIXx SystemBuild[§§§] tool . Each axis is a separate superblock and superblock elements are used for the integrator and other major elements. Once the model was tested the Ada code is generated separately for each axis; i.e. each axis is contained in a separate parent superblock. Of course , the code could not be targeted for the host simulation system but the Ada code for the model would be packaged in such a way that it could be lifted out and interfaced to the host system code. Initially the Ada code was developed on a VAXstation[****] computer using a non-real-time test program. The code had to be modified to be simulated and tested.

The actual procedure of converting to a stand alone package was straight forward; however, it became apparent that after being rearranged, having utility procedures added, transported to the simulator development system, and being modified to fit the simulation architecture that the code would bear little resemblance to the original code. Because of the extent of the metamorphosis, changes to the original model could not be made through the original block diagrams then autocoded. The code would have to stand alone and therefore would have to be readable. Once complete, the final code had little resemblance to the original code that was generated automatically.

An abbreviated copy of the automatically generated code is shown in Figure 7. The procedure SUBSYSTEM_1 contains the entire elevator surface simulation The code is readable to a degree and heavily commented. Each comment line references code directly to an element in the block diagram. Using the block diagram of the model it is easy to find in the code where an element is simulated. Without the diagram this code would in my opinion be difficult to read and modify. The variable names are structured not arbitrary but cryptic looking. To a degree the model builder has control over the names by naming the elements and signal paths with meaningful names which would be reflected in the code.

The location of code for each element appeared scrambled and did not flow well. All the code within a parent superblock is made in-line and reusable functions and procedures are not used even for the standard elements. Also, the code for the dynamic elements are duplicated and the state data is passed outside the package rather than being saved internally. Constant data is not declared as such, rather, it is loaded into a local variable each pass. The linear interpolation function is the only reusable procedure, but the source code is not available so it cannot be

```
separate (SUBSYSTEMS)
  PACKAGE BODY SUBSYSTEM_1_PKG is

      procedure SUBSYSTEM_1 is
         o
         o
         o
       RT_PCONT_1           : RT_FLOAT renames BUS(17);
       RT_PCONT_2           : RT_FLOAT renames BUS(18);
         o
         o
         o
       begin
         o
         o
         o
   ------------------------------------------------------------ limited integrator
   -- (ELEV_PCU..24)
      LIMITED_POSN := ELEV_PCU_24_1 + 0.025 * ELEV_PCU_7_1 ;
      if ELEV_PCU_7_1 > 0.0 and  LIMITED_POSN  > 15.0 then
            LIMITED_POSN := 15,.0 ;
      elsif ELEV_PCU_7_1 < 0.0 and LIMITED_POSN < -40.0 then
            LIMITED_POSN := -40.0 ;
      end if ;
      LIMITED_POSN := 1.0 * LIMITED_POSN ;
   ------------------------------------------------------------ general nested expression
   -- (rt_trim switch..86)
      RT_TRIM_SWITCH_86_1 := 0.0
             o
             o
             o
       ------------------------------------------------------------linear interpolation
       -- (rt_pcont.stiction.26)
         linear_interp(BUS(34..34),BUS(67..67),1,R_P(6..25),I_P(1..8)
                                         );
       ------------------------------------------------------------gain block
       --(rt_pcont.Bdq.14)
        BDQ_1 := 20.0 * S1_1_1;
             o
             o
             o
        ------------------------------------------------------------ limited integrator
        -- (ELEV_PCU..24)
         ELEV_PCU_24_1_B1 := ELEV_PCU_24_1 + 0.025 * ELEV_PCU_7_1 ;
         if ELEV-PCU_7_1 . > 0.0 and  ELEV_PCU_24_1_B1 > 15.0 then
              ELEV_PCU_24_1-B1 := 15.0 ;
         elsif  ELEV_PCU_7_1 < 0.0 and ELEV_PCU_24_1_B1 < -40.0 then
              ELEV_PCU_24_1_B1 := -40.0 ;
         end if ;
              o
              o
              o
        end SUBSYSTEM_1_PKG ;
```

Figure 7 Sample of Autocode Generated Software

[†††] AutoCode is a registered trademark of Integrated Sysyems, Inc.

[‡‡‡] MATRIXx is a registered trademark of Integrated Systems, Inc.

[§§§] MATRIXx and SYSTEMBUILD is a registered trademark of Integrated Systems, Inc

[****] VAXstation is a registered trademark of Digital Equipment Corporation

transported.

Automatically generating Ada code did not give a 10 to 1 or even a 5 to 1 improvement in productivity because of the transition the code underwent; however, I think it was and still can be of great importance in the development process in the following ways and especially if the following improvements are made.

(1) Given our lack of experience with Ada and the host simulation architecture autocoding gave us a quick start in delivering correct, testable, executable code. The conversion to code that could be tested in non-real-time on the VAXstation was only a few hours per axis.

(2) Once the code was verified it could begin a step-by-step conversion to its final form. It did not take long to convert the code to a form that would compile, link, and execute on the host simulation. The remainder of the development has to do with style and OOD. These improvements could be done in a more relaxed pace because we had executable code if needed.

(3) The present state of autocoding is best used in rapid prototyping where the long-term life-cycle of the code is not important. The final version of the autocode can be used as an executable specification for the final version that is to be developed.

(4) An autocoding tool has to be directed more to a general user host system and not a particular prototyping system. The use has to be able to specify the interface and to insert their own custom utilities. More care should be taken to partition the subsystems into superblocks and the superblock code should be in separate procedures. The user needs to take care to the naming of the signals and the element names so that the variable names in the code are meaningful.

(5) The forethought that the user has in the development process and the options they have available will streamline the development process from the the auto-generated to the final form that fits the style and software architecture of the host system.

Conclusions

Although Ada is proclaimed as the DOD's embedded software high level language, it can have a great impact on non-embedded software also. Even though this has been said before it needs to be said again and again because there is still resistance to using Ada over FORTRAN for general purpose programming and real-time simulation. The object-oriented approach works well with Ada for applications to simulation of avionics and flight control systems in a multiprocessing environment. Using an automatic code generator to create software from control law block diagrams is feasible and the code generated is readable, but the process still has far to go with respect to tailoring the software to specific host systems. The auto-generated code gives a jump start for code development and requires no additional labor since it is automatically generated from the block diagram model needed for development and testing .

References :

1. C+C++ Programming With Objects in C and C++ by Allen Holub
McGraw-Hill, Inc 1992 .

2. ***No Silver Bullet*** Essence and Accidents of Software Engineering
Frederick B. Brooks, Jr.
IEEE Computer Magazine , April, 1987 .

AIAA-94-3409-CP

777 SYSTEMS INTEGRATION LAB (SIL) ARCHITECTURE OVERVIEW

R. C. Kircher, Jr.
Flight Systems Laboratory
The Boeing Company
PO Box 3707 MS 19–HH
Seattle WA 98124
(206) 662–4834

Abstract

The Boeing 777 program Service Ready initiatives established the need for a Systems Integration lab (SIL). Airplane–level systems testing would be conducted in the SIL. To support the SIL test requirements and maintain compatible with the 777 standalone test stations, a unique Lab hardware and software system architecture was developed. This paper provides an overview of the SIL hardware and software architecture.

Why the 777 Had a SIL

The 777 program established fourteen major program initiatives at its inception, one of which was "Deliver a Service Ready Airplane". To support this initiative, Boeing Commercial Airplane management committed to the development and use of a Systems Integration Lab (SIL). The goal of the 777 SIL is to ensure that 777 electrical/electronic systems work as an integrated whole. In prior programs, e.g. 757/767, 747–400, the airplane systems were tested in their standalone test stations with some limited integration testing. Final systems integration was accomplished on the flight test airplanes. The 777 SIL was designed for airplane–level integration testing of all major systems while using production equivalent airplane wiring with airplane quality electrical power. See Fig. 1.

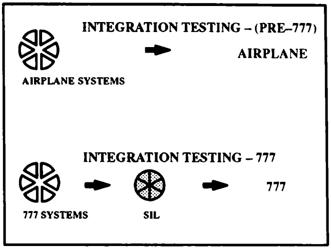


Fig. 1. Comparison of Integration Test Methods

SIL Test Strategy – Overview

In an overview sense, the SIL test strategy was: infiltrate 777 electrical/electronic systems into the SIL as they complete their standalone testing; conduct airplane–level validation testing after all the systems have been infiltrated. Demonstrate a "working airplane".

The two major SIL test configurations are Static and Dynamic. In the static configuration, the "SIL airplane" is parked on the ramp. Basic system checks may be performed, e.g., Auxiliary Power Unit (APU)/engine starts, hydraulic system checks, Environmental Control System (ECS), pilot controls' sweeps, etc. The Dynamic Configuration (Fig. 2.) supports taxi and flight testing. The Static Configuration is identical to Dynamic except for ADIRS (Air Data Inertial Reference System), Radio Altimeters, and Navigation systems (ILS, VOR, DME, ADF/MB). In the Dynamic configuration, those systems are simulated; the real hardware is used for Static testing.

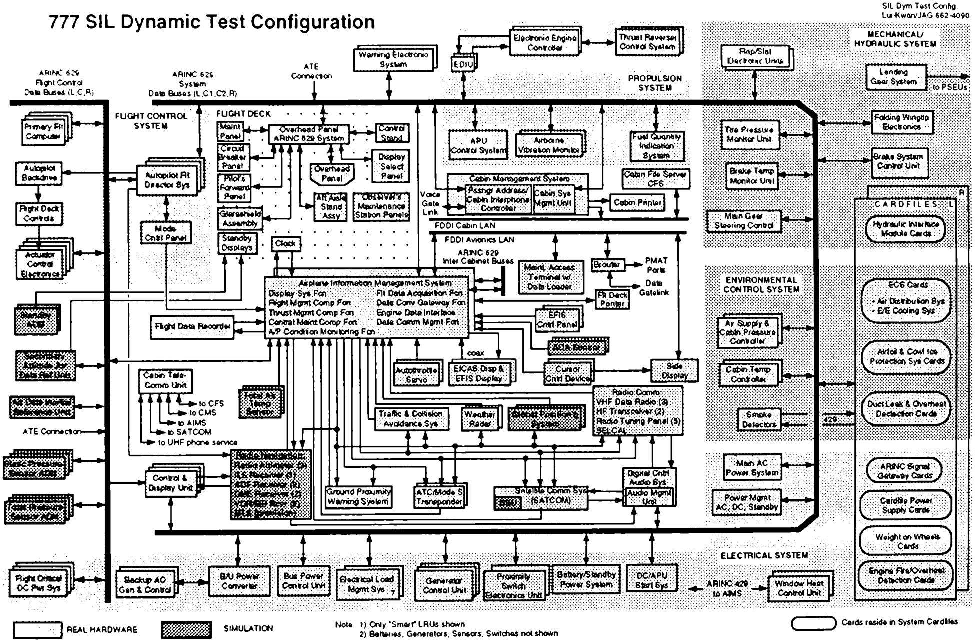


Fig. 2. 777 SIL Dynamic Test Configuration

Implicit in the SIL test strategy is the use of 1) real 777 engine/APU electrical power generators, and 2) 777 Flight Test instrumentation and data recording systems.

SIL Lab Requirements derived from the Test Strategy

Infiltration/Integration Testing:

Because of 777 inter-system dependencies and member system development schedules, at certain times during the infiltration schedule, simulated LRUs would be needed. The simulated unit would be replaced by the real LRU when that hardware completed standalone testing. To allow for unplanned adjustments to the infiltration schedule, the lab needed to provide "plug compatible" simulations for approximately 28 of the 777 systems. The requirement for plug compatible real and simulated LRUs was especially challenging because of the unpredictable numbers of real/simulated combinations and the resulting complexity of simulator dataflow.

To ensure the smooth transition of LRUs from standalone testing to SIL integration testing, the SIL and standalone "simulation environments" needed to be as compatible as possible. This would minimize the number of test parameter changes thereby facilitating troubleshooting during the engineering test program.

SIL Static/Dynamic Test Configurations:

Switching between SIL static and dynamic test configurations would require the same "real/simulated" swapping of ADIRS, Radio Altimeter, and Navigation systems as needed during Infiltration testing.

Airplane Electrical Power:

Because the SIL would use a real 777 Electrical System, including ELMS (Electrical Load Management System) and IDGs (Integrated Drive Generators), the lab would need to provide motor/generators and real-time control systems to drive the IDGs at the rotational speeds of the simulated left and right engines and APU.

SIL Instrumentation:

To facilitate the support of the Flight Test program, the SIL needed to be instrumented with the same system as used in the Flight Test airplanes. Because many systems would be simulated in the SIL, the flight test instrumentation system would need to be enhanced to allow the recording and play back of simulator-internal (memory) parameters.

SIL Architecture Challenge

SIL Challenge: Provide a system architecture that supports the testing strategy planned by the customer. In particular, a system architecture that:

1) is compatible with the 777 standalone test stations,

2) provides the simulator modularity and flexibility to quickly reconfigure from any of approximately 28 simulated to real LRUs and vice-versa,

3) supports real time data communication with the engine/APU IDG motor/generators and the SIL/Flight Test instrumentation system.

SIL Computing System Architecture

The SIL system architecture is based upon technology developed by FSL and ESL for use in their standalone test facilities.

The PSIM (Parallel Simulation) H/W and S/W simulation technology is used in the SIL and all 777 FSL standalone test stations (Flight Controls, Propulsion, Avionics/AIMS (Airplane Information Management System), Flight Deck) and Cabs. For these systems, the SIL maintains "component-level", as opposed to test station-level commonalty with those test stations and Cabs.

The ECS, Mechanical/Hydraulic and Electrical System standalone test stations employ the Electrical Systems Lab (ESL)-developed ETS-200 simulation technology. Early in the project it was decided that the ETS-200 test stations as well as their LRUs would be part of the SIL. In these airplane systems, the SIL maintains test station-level commonalty with the standalone facilities.

An overview of the computing architecture is shown in Fig. 3.

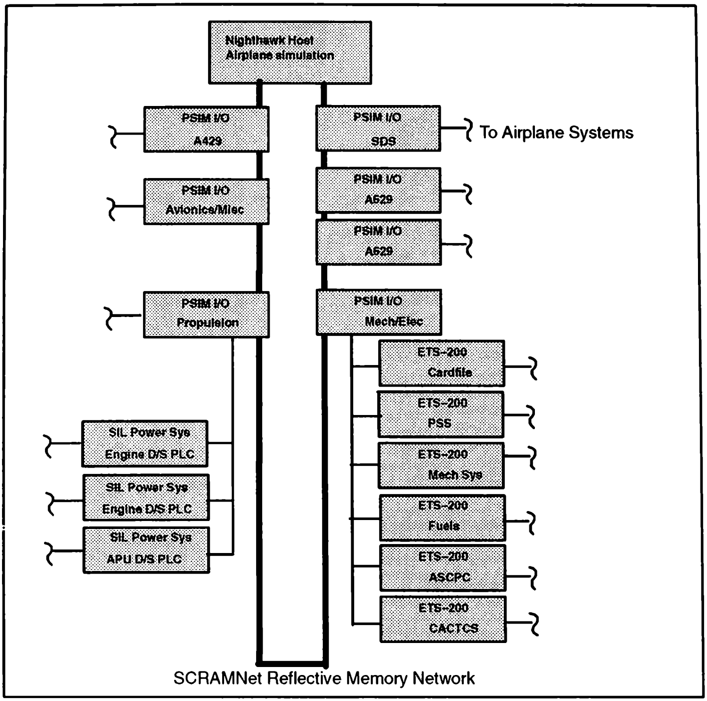


Fig. 3. SIL System Architecture Overview

The main components of the SIL computing system architecture are

A) the Harris 8-CPU Night Hawk-based PSIM host simulation computer,

B) six dual-CPU Concurrent-based ETS-200 simulation systems,

C) the Mech/Elec node,

D) four PSIM I/0 nodes,

E) the SIL Data System, and

F) the SIL Power System.

A fibre optic, real-time reflective memory network (SCRAMNet) provides a common memory window to the Night Hawk, the PSIM I/0 nodes, the Mech/Elec node, and the SIL Data System (SDS). The SCRAMNet and Bit 3 connections to the Mech/Elec node provide real-time data communication between the PSIM-based simulation system and each of the six ETS-200-based simulation systems. A General Electric Genius LAN system provides communication between the Propulsion PSIM I/0 node and the SIL Power System Programmable Logic Controllers (PLC). The engine/APU drivestands and IDGs, located in an adjacent room, are controlled by the PLCs.

SIL Simulation S/W Architecture

The principle SIL software architectural elements include the PSIM airplane simulation S/W, six ETS-200 Subnode environment simulations, SIL Service Layer and Utility software, DMS (Device Management System) I/O files, and Power System S/W, and the SIL Data System S/W. See Fig. 3.

The majority of the PSIM simulation S/W models and DMS I/O files are developed by the FSL 777 Common Models organization, which provides S/W for use in all FSL Test Stations (Flight Controls, AIMS, etc.) and simulator cabs. The Common Models organization, working closely with 777 design engineers, develop this S/W for initial use in the Crew Training Simulator Checkout documents as well as the LRU SCD (Specification Control Drawing) documents. The S/W model interfaces, as well as the DMS I/O scaling S/W, are derived directly from the 777 ICD (Interface Control Document) to ensure compatibility. These models are linked with SIL-unique software models to build the executables that run in the Lab. When connected with the DMS I/0 systems, the PSIM simulation provides the SIL with 777 LRU simulation functionality as well as environment or sensor signals for real LRUs. Because each of the FSL standalone test station simulations derive from the same Common Model source, software compatibility among the 777 test stations and the SIL is ensured.

ETS-200 simulation software provides environment signals and I/O for ESL supported LRUs. The ETS-200 simulations range in functionality from I/O pass-through to full fledged plant models. The ESL standalone test station organization provides the original software from which SIL-unique changes are made, thereby maintaining functional commonalty with the stand alone labs. In the standalone labs, airplane state vector, e.g. altitude, airspeed, and so on, are manually driven by the user. In the SIL, those signals are calculated in the PSIM airplane simulation and communicated to the ETS-200 systems via SCRAMNet memory at the Mech/Elec node.

SIL Service Layer software executes on both the Night Hawk and Concurrent systems and ensures the coordinated startup and real-time execution of both simulation systems. On startup, this software initializes the area of SCRAMNet memory that each of the ETS-200 subnodes shares with the PSIM simulation. After startup the Service layer S/W ensures that the ETS-200 simulation modes (IC, Compute, ...) track those of the PSIM simulation. The Service Layer software also monitors and graphically portrays the system states to facilitate system diagnostic troubleshooting.

The SIL Power System software provides the manual and automatic control of the SIL Power Room engine and APU drive stands and electrical loading systems. In the automatic mode, PSIM simulated engine and APU speeds (N1, N2, etc.) are communicated to the drive stand controllers through PSIM I/0 links and the Genius-LAN computing systems. The drive stands (simulating actual engines drive shafts) turn the 777 Integrated Drive Generators (IDG) thereby supplying power to the ELMS (Electrical Load Management System) for distribution to the SIL LRUs and systems.

The software coupling relationships are shown in Fig. 4.

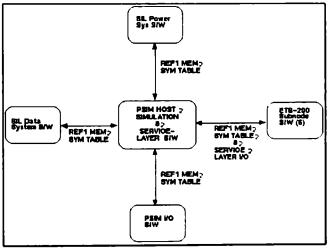


Fig. 4. SIL Software Coupling

SIL Simulation S/W Development

The PSIM simulation S/W development employed a phased approach:

Phase 1 – Interface only simulations,

Phase 2 – integrated airplane simulations supporting either 777 propulsion or flight controls SIL testing but not both at the same time, and

Phase 3 – simulations supporting all SIL systems.

Fig. 5. is a stylized portrayal of SIL S/W development as it relates to the infiltration of 777 LRUs into the SIL.

Phase 1:

Interface–only simulations included only the symbol table (memory map) of the full airplane simulation, and the companion signal I/O scaling table files. The symbol table files as well as the DMS I/O files derive from 777 ICD database. Simulation models were not included. Users controlled the value of the simulation output signals through various PSIM utilities.

Interface–only simulations initially were also used for integration testing of the PSIM/ETS–200 "marriage" and laboratory signal continuity testing.

These simulations were used by Lab personnel to check signal continuity from the host computer (measured in engineering units) to the LRU rack/shelf interfaces (measured in volts or digital bit patterns) in the SIL. In addition to finding wiring problems, this testing also surfaced I/O scaling and other errors in the DMS I/O files that were corrected prior to LRU infiltration testing. These same simulations were also used for those LRU infiltration tests not requiring simulator model functionality, i.e., where manual control of simulated signals was sufficient.

Phase 2:

During this time period the majority of SIL LRUs were infiltrated. The complexity of these systems' interfaces required high fidelity simulations. 777 Common Model and SIL–unique model functionality were introduced into the SIL during Phase 2.

Because of the fast control loops inherent in propulsion and flight controls systems, the "standalone" nature of the SIL testing at that time, and Night Hawk performance capabilities, two separate development and testing paths were followed; one supporting Propulsion system testing and one for Flight Controls.

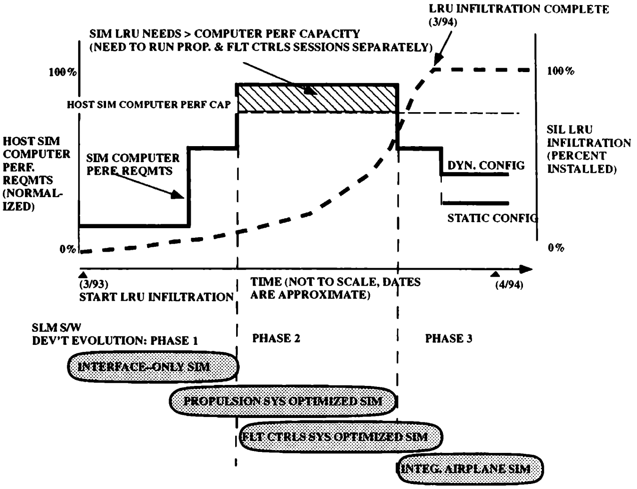
In the "propulsion path" simulation, the Common Models were partitioned in the host processors to accommodate the engine control loops at the expense of the flight controls loops. The "flight controls development path" did the opposite. During this time period the majority of the LRUs in the SIL were still being simulated.

Because these two configurations could be tested separately, this limitation did not impede 777 SIL test progress

Phase 3:

As the final LRUs were infiltrated into the SIL, replacing their simulated counterparts, the Night Hawk computing load was reduced, thereby allowing the simulation engineers to optimize both the propulsion and flight controls loops and eliminate the dual development path.

As of the writing of this paper, and except for AOG (Aircraft on Ground) workaround simulations, only one simulation path is required.


LRU INFILTRATION COMPLETE
(3/94)
SIM LRU NEEDS > COMPUTER PERF CAPACITY
(NEED TO RUN PROP. & FLT CTRLS SESSIONS SEPARATELY)
100%
HOST SIM COMPUTER PERF CAP
HOST SIM COMPUTER PERF. REQMTS (NORMALIZED)
SIM COMPUTER PERF REQMTS
DYN. CONFIG
STATIC CONFIG
100%
SIL LRU INFILTRATION (PERCENT INSTALLED)
0%
0%
(3/93)
START LRU INFILTRATION
TIME (NOT TO SCALE, DATES ARE APPROXIMATE)
(4/94)
SLM S/W
DEV'T EVOLUTION: PHASE 1
PHASE 2
PHASE 3
INTERFACE-ONLY SIM
PROPULSION SYS OPTIMIZED SIM
FLT CTRLS SYS OPTIMIZED SIM
INTEG. AIRPLANE SIM

SIM S/W EVOLUTION AS A FUNCTION OF SIL LRU INFILTRATION
AND THEIR IMPACT ON HOST SIMULATION COMPUTER PERF. REQMTS

Fig. 5. SIM S/W Evolution

I/O SYSTEMS FACILITATE THE DEVELOPMENT AND CERTIFICATION OF NEW COMMERCIAL AIRPLANES

Joseph W. Ortiz
PO Box 3703 MS 19–MJ
Seattle WA 98124
(206) 662–4756

Abstract

The I/O system is a data highway used for providing the airplane signal responses needed to interface the simulation to the Airplane Line Replaceable Modules (LRM) workstations or engineering simulator (sometimes referred to as the cab). The complexity of modern airplanes has led to more extensive testing performed in the lab to support subsystem integration and certification of airplane devices. This especially applies to the Boeing 777, which is presently under development. This new I/O system is designed to support the advanced avionics design, the expanded involvement of the end user, and the advances in Flight Systems Lab (FSL) computer technology and environment. This paper describes the I/O system used by FSL organization of the Boeing Commercial Airplane Group.

Introduction

An I/O system is defined to consist of a host(s) manipulating the Simulation variables required by the LRMs, workstations and engineering simulators and real–time network for transferring simulation variables from the host to the LRM's and the diagnostic software.

With the development of the 777, came additional advanced technology to be used on Boeing Commercial airplanes. This technology has forced FSL's I/O systems to become larger and more complex. Arinc 629 is one of the new developments that transfers a large number of variables on the digital data bus at high speeds. Technology like this has forced FSL to take a new approach to I/O system design.

The concept is called reflective memory. All computing resources are connected to a high speed fiber optic network called (Shared Common Random Access Memory Net) SCRAMNet (trade mark of the Systran Corporation). One or more of the computed resources will connect to the workstation or cab.

When information is put on the fiber optic network, it is reflected to all systems on the SCRAMNet bus.

The 777 Engineering Simulator #2 I/O system contains 16 computing systems. Each system contains a SCRAMNet real– time interface card that is connected to the SCRAMNet bus (see CAB2 System Architecture Fig. 1. thru Fig. 3.). These systems are:

VME I/O Node Controller Devices:

1. Arinc 629 System Bus
2. Arinc 629 Flight Controls Bus
3. Arinc 629 Cross Cabinet Bus
4. Arinc 429 Bus 1
5. Arinc 429 Bus 2
6. I/O Device System Bus
7. Night Hawk 4807 (host)
8. Control Loader
9. Map Display (SGI)

10.–12. Visual System (CGI)
(3 SCRAMNet cards)

13.–16. Digital Display System (DDS)
System 1 through 4

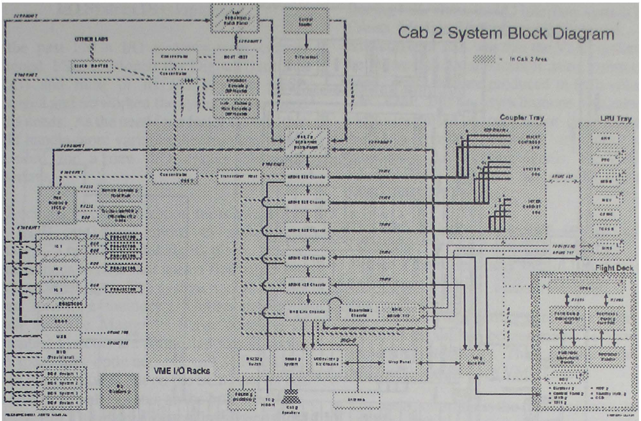


Fig. 1. Cab II System Block Diagram

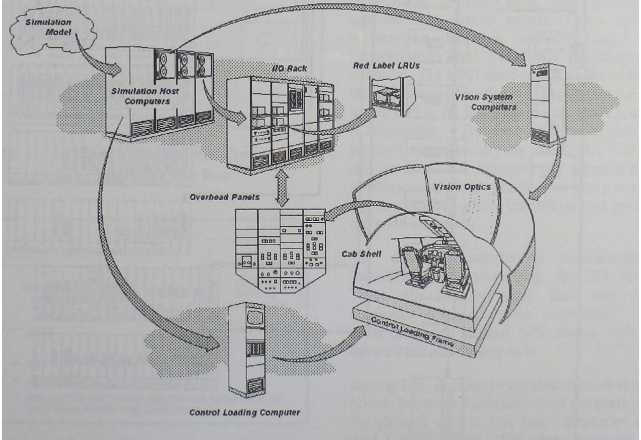


Fig. 2. Cab II System Configuration

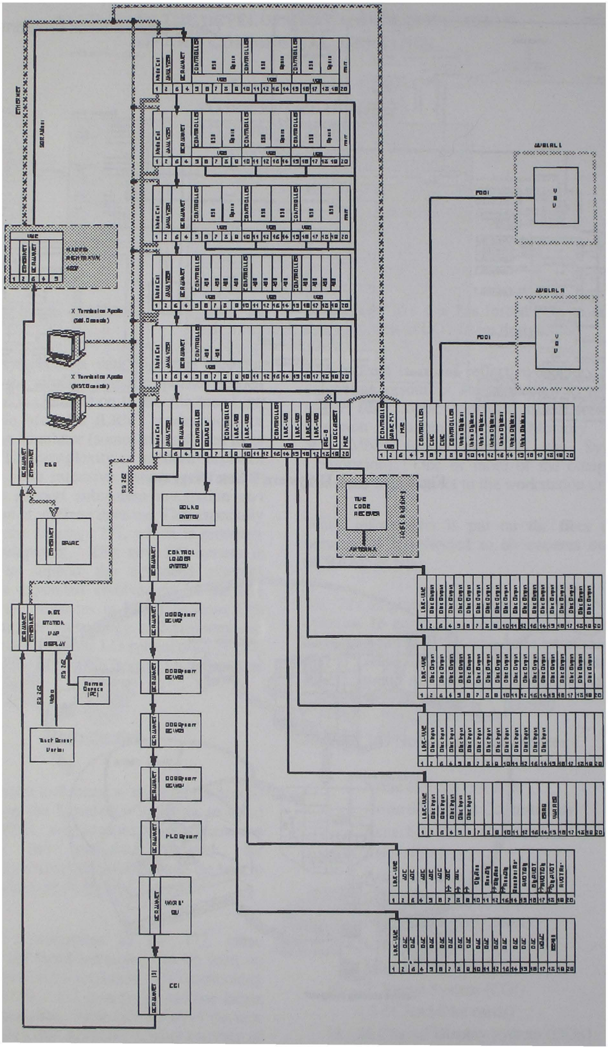


Fig. 3. 777 Cab II System Architecture

I/O System Development

In the past FSL's I/O systems were custom designed. FSL developed the Digital Data Bus (DDB) and most of its I/O devices. FSL packaged and networked the system customized for its needs. As the need for a faster system, that could handle more variables became important, the need for a new I/O system also was important.

As Boeing's FSL organization turns away from the DDB and turns more to commercial standards, a new I/O is taking its place. It is an asynchronous system that uses a fiber optic bus to transport its data. The system uses a concept called reflective memory. Unlike the DDB which uses the Master to Slave concept, all computing systems are connected to a common high speed fiber optic network call SCRAMNet. When information is put on the SCRAMNet bus it is reflected to all systems on the bus (see Fig. 4. and Fig. 5.). This type of system has a bandwidth of 1 million words per second. The DDB I/O system achieved a speed of 500,000 words per second.

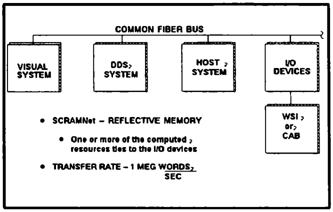

COMMON FIBER BUS
VISUAL SYSTEM
DDS SYSTEM
HOST SYSTEM
I/O DEVICES
WSI or CAB
• SCRAMNet – REFLECTIVE MEMORY
• One or more of the computed resources ties to the I/O devices
• TRANSFER RATE – 1 MEG WORDS/SEC

Fig. 4. All Computing Systems Connected to a Common Fiber Bus

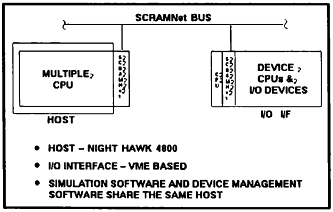

SCRAMNet BUS
MULTIPLE CPU
SCRAMNet
HOST
CPU
SCRAMNet
DEVICE CPUs & I/O DEVICES
I/O I/F
• HOST – NIGHT HAWK 4800
• I/O INTERFACE – VME BASED
• SIMULATION SOFTWARE AND DEVICE MANAGEMENT SOFTWARE SHARE THE SAME HOST

Fig. 5. One or More Computing Systems Talks to the I/O Interface

Boeing FSL chose its I/O interface system to be (Versa Module Eurocard) VME based. The advantage for FSL was that the VME system can be tailored to an application using commercial parts. Parts that are produced in high volumes. The VME Bus is asynchronous (meaning no clocks are used to coordinate data transfers). Data is passed between modules using interlocking handshaking signals. The VME bus handles data transfers at speeds up to 8 MegaBytes per second.

The VME I/O interface system contains a Node Controller, a SCRAMNET interface card, Device Controllers and the I/O device cards located in a chassis.

The SCRAMNet interface card was chosen from Systran Corporation because of its performance, growth and flexibility. This card sits in the third and fourth slot of each VME chassis.

The node controller is the first card in each VME chassis. The node controller is responsible for initializing and controlling the SCRAMNet memory, listening for and responding to host control and for being the interface to the host for device controllers. This interfacing includes symbol table lookups: setting, clearing and relaying SCRAMNet interrupts: host control and acknowledgement; passing error information to the Host and polling for non–interrupt driven events. Other software running on the node controller target CPU includes node level VACCESS and Diagnostics software. The CPU itself serves as the arbiter for the node and as the ethernet gateway for the other target CPUs in the node using the backplane configuration. The first of these functions is handled automatically by the CPUs hardware and the second is transparently handled by the VxWorks operating system. It is also possible to run low priority device controllers and model on the node controller.

The device controller is responsible for initializing and controlling the I/O devices. There are usually more than one device controller for each node controller. These controllers are labeled 629 device ctrlr, 429 device ctrlr and analog ctrlr.

Boeing FSL decided to use commercial standard boards because of availability of the parts and of the cheaper cost to buy large quantities rather than building them. FSL does however build some of the device boards used. Arinc 629, Master Clock Reset, Sound System, and VSB/VME interface expander cards are some of the cards being built by Boeing. These particular cards were not available from commercial companies at this time. For example, the 777 airplane selected the Arinc 629 digital bus as its primary communications bus. During simulation development, FSL was the only source of Arinc 629 hardware interfaces. The advantage of building the Arinc 629 card was that the simulation design could be tailored based on different customer group requirements. These groups are Flight Deck, Aerodynamics Research, Airplane Information Management System (AIMS) test bench and the Flight Controls groups of FSL.

One method of expanding a node is to use the FSL built VSB to VME Link cards. The slots labeled VME/VSB link represent an external connection to another VME expansion chassis containing I/O cards. The connection components include a VSB slave card, VME master card and a cable. The VSB slave card translates VSB bus read and write requests to the VME master card in the expansion chassis across the external cable. The VME master then arbitrates to complete the request on the VME bus. Bus requests in the opposite direction are not possible. The 777 Engineering Simulator #2 uses this card to expand the analog I/O devices (see Fig. 3.).

It is also possible to expand the size of a node using bus extenders such as the Hal–Versa. VME bus bandwidth and reliability limit this type of expansion. Bit–3 VME/VSB extenders are another method of expanding the size of a node. Like the VME/VSB connections, bus request are one way.

Each I/O system created has a system I/O configuration file associated with it. The graphics based tool, I/O Configuration Editor (ICE), allows the bench builder to create and maintain a data base of information that completely defines each piece of equipments I/O interface. This information is organized by node, target CPU, and controller. It defines such things as addresses, available channels, default settings for I/O scaling assignments, card types and models. This data base is compiled on command into a binary system configuration file that is read by the Device Management Software (DMS), VACCESS and various other diagnostic configuration controlled directories on the hosts.

The system configuration file is also used by the DMS user interface to verify I/O assignments, supply default parameters and to provide addressing for connecting with the device controllers of the system.

FSL Networking

The interactive networks within FSL utilize Ethernet for file transfer and remote login. There are currently over 2200 nodes served by Ethernet, including compute servers, workstations, x–terminals, Bench/Cab I/O systems, and terminal servers.

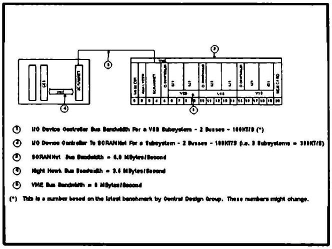
The Ethernet backbone consists of four interconnected Cisco routers providing thirty–four LAN segments. Connection to end nodes is via seventy–five Synoptic Lattisnet concentrators, located in six main data–closet wiring hubs, and seventeen Lab locations where the densities of nodes required concentrators. The type of Synoptic equipment used has been evaluated and approved by Boeing Computer Services, and is a standard throughout Boeing. Every concentrator contains an intelligent network management module that communicates with two network management consoles. Via the software programs SunNet Manager and Lattisnet Network Management for Unix, the network administrator has full visibility of network traffic and conditions on all Lan segments, and can take action to eliminate problems before they occur. A similar network management system exists for the 250 HP–Apollo Domain–Ring workstations within FSL.

The real–time networks within FSL are used for communication of data in real–time between nodes simulating flight conditions of an aircraft. The networks are based on a reflective memory system manufactured by the Systran Corporation. Nodes in a simulation are arranged in a ring call a Gold ring, for guaranteed access, optimized, low overhead and deterministic. Typical nodes in a gold ring cab, a DDS (Digital Display System) and a CIG (Computer Image Generator).

Each node in the ring contains a SCRAMNet board, loaded identically. When any node updates a location in SCRAMNet memory, the new value is compared to the old value, and if different, is transmitted to the other nodes in the ring at 150 megabits per second to update their memories. Currently, twenty–six real–time simulations can run simultaneously.

Airplane Simulation I/O Loading Requirements

A benchmark measurement done by FSL showed that each node controller's bus bandwidth could handle a maximum load of 300,000 words per second (see Fig. 6. and Fig. 7.). This meant that the total load amount of the device controllers in that node could not exceed 300,000 words per second. Knowing that ARINC 629 would contain worse case loading, FSL set a standard for an ARINC 629 SubSystem. The system would contain a maximum of three device controllers allowing each to have a load maximum of 100,000 words per second. This was also true for the ARINC 429 SubSystems. All other SubSystems depending on the configurations could contain as many device controllers as needed. The limitation being the node controllers loading capacity of 300,000 words per second.



① I/O Device Controller Bus Bandwidth For a V80 Subsystem - 2 Busses - 100KT/S (*)

② I/O Device Controller To SORAMNet For a Subsystem - 2 Busses - 100KT/S (i.e. 3 Subsystems = 300KT/S)

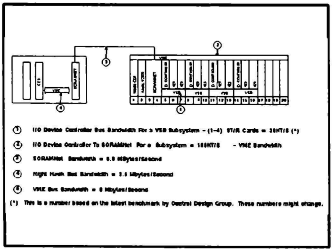
③ SORAMNet Bus Bandwidth = 6.0 MBytes/Second

④ Night Hawk Bus Bandwidth = 3.6 MBytes/Second

⑤ VME Bus Bandwidth = 8 MBytes/Second

(*) This is a number based on the latest benchmark by Central Design Group. These numbers might change.

Fig. 6. System Specifications (For an ARINC 629 SubSystem



① I/O Device Controller Bus Bandwidth For a V80 Subsystem - (1-4) ST/R Cards = 30KT/S (*)

② I/O Device Controller To SORAMNet For a Subsystem = 100KT/S - VME Bandwidth

③ SORAMNet Bandwidth = 6.0 MBytes/Second

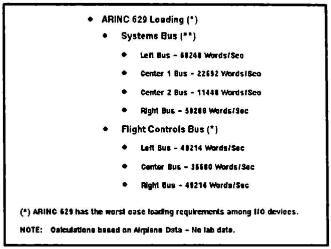
④ Night Hawk Bus Bandwidth = 3.5 MBytes/Second

⑤ VME Bus Bandwidth = 8 MBytes/Second

(*) This is a number based on the latest benchmark by Central Design Group. These numbers might change.

Fig. 7. System Specifications (For an ARINC 429 SubSystem

To assure that the VME I/O interface system could handle the 777 Cab2 I/O requirements, FSL calculated the amount of airplane I/O data that the system would need to support. It was determined that Arinc 629 contained the largest case for loading among the I/O devices. It would need to handle a total load of 496,040.80 words per second for the eleven busses supporting it. Arinc 429 required a total loading of 253,000 words per second for the one hundred and twelve busses supporting it. FSL knew from experience that the I/O loading requirements for the analog devices was approximately 50,000 word per second. A small load that would have little impact on the I/O system so FSL decide it would use the estimations. Loading requirements for these devices are broken out in more detail and are located on Fig. 8. and Fig. 9.

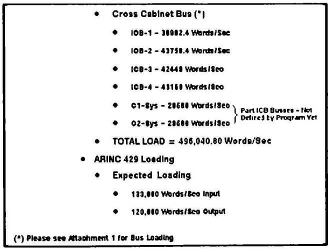


- ARINC 629 Loading (*)
  - Systems Bus (**)
    - Left Bus - 60240 Words/Sec
    - Center 1 Bus - 22692 Words/Sec
    - Center 2 Bus - 11448 Words/Sec
    - Right Bus - 58288 Words/Sec
  - Flight Controls Bus (*)
    - Left Bus - 48214 Words/Sec
    - Center Bus - 36680 Words/Sec
    - Right Bus - 48214 Words/Sec

(*) ARINC 629 has the worst case loading requirements among I/O devices.

NOTE: Calculations based on Airplane Data - No lab data.

Fig. 8. Loading Requirements for Key Devices



- Cross Cabinet Bus (*)
  - ICB-1 - 38982.4 Words/Sec
  - ICB-2 - 43758.4 Words/Sec
  - ICB-3 - 42448 Words/Sec
  - ICB-4 - 43168 Words/Sec
  - C1-Sys - 20680 Words/Sec
  - C2-Sys - 20680 Words/Sec

  } Part ICB Busses - Not Defined by Program Yet
- TOTAL LOAD = 496,040.80 Words/Sec
- ARINC 429 Loading
  - Expected Loading
    - 133,000 Words/Sec Input
    - 120,000 Words/Sec Output

(*) Please see Attachment 1 for Bus Loading

Fig. 9. Loading Requirements for Key Devices (cont'd)

What the results show is that the Arinc 629 Systems Bus loading is equal to 143,496 words per second on the node. This means that about forty seven percent of the node controllers capacity is being used. Arinc 629 flight controls bus loading is equal to 135,108 words per second on the node. This means that about forty five percent of this Node controllers capacity is being used. Arinc 629 cross cabinets bus loading is equal to 168,300 words per second on the node. This means that about fifty six percent of the node controllers capacity is being used. Since the analog loading was not calculated and was assumed to be 50,000 words per second, only seventeen percent of the node controllers loading capacity is being used.

I/O System Performance

The 777 CAB2 I/O system contains 16 computing systems. Below are the results of each systems loading requirements and the total load requirement needed to be on the fiber optic bus. The information below was provided by various groups within the FSL organization.

VME I/O Node Controller Devices:

1. Arinc 629 Systems Bus –	.6 MegaBytes/second
2. Arinc 629 Flight Controls Bus –	.6 MegaBytes/second
3. Arinc 629 Cross Cabinet Bus –	.6 MegaBytes/second
4. Arinc 429 Bus 1 and	
5. Arinc 429 Bus 2 –	.125 MegaBytes/second
6. I/O Device Bus –	(assumed) .02 MegaBytes/second
	Total = 1.945 MegaBytes/second
7. Control Loader –	.8 MegaBytes/second
8. Map Display (SGI) –	.042 MegaBytes/second
9. Visual System (CGI) –	.012 MegaBytes/second
10. DDS System 1 through 4 –	.030 MegaBytes/second
	Sum Total = 2.829 MegaBytes/second

The SCRAMNet fiber optic bus has a capacity of 6.8 MegaBytes per second. The results show that it could handle the system loading capacity with approximately fifty percent growth. The host capacity has a capacity of 3.5 MegaBytes per second. The results also show that the host Night Hawk could handle the system loading with approximately ten percent growth. The VME bus has a capacity of 8 MegaBytes per second. The results also show that less than five percent of the VME bus bandwidth is being used. The results are based on worst case loading and is assumed that the 777 CAB2 simulator will not contain all the loading variables.

FSL Lessons Learned

By buying our VME boards off the commercial shelf the 777 CAB2 hardware design group found that money could be save by buying in large volumes. A problem with an off the shelf board is that if the board purchased has unknown problems, as did an analog discrete output board. It had an isolation problem which caused FSL to re–design the board as the Vendor was unable to provide a solution. Thirty six boards had to be modified at a cost to the 777 Cab2 program hardware design group of approximately three thousand six hundred dollars.

The total cost for the new I/O system interface equals to seven hundred and fifty thousand dollars. More than ninety percent of the system contains commercial parts. The dollar value for the Cab2 I/O system is small compared to the quality of system that is in place and for what it will do in the future for the 777 airplane program.

The VME I/O interface system was up and running March 1,1993. The system up and ran with very few problems. These were the major problems.

1. Arinc 429 VSB handshake error on board.
2. Arinc 629 switched memory problems on board.
3. Discrete output isolation problem on board.
4. CPU board failed after running two weeks.
5. Irig–B card failed few weeks later.

Because the Cab2 I/O system was running on time, problems 1 through 3 were fix by the FSL Central design group using it as the test bench. It helped FSL because the lab benefited from the solutions. Problems 3 and 4 were replaced with other boards and were sent back to the vendor to be fixed.

The I/O interface system is designed to be modified or expanded using a small degree of configuration change. It also has been design for a thirty percent growth expansion capability.

Conclusion

The I/O system has gained acceptance by users in the Flight Systems Lab and by the new Boeing 777 airplane program. Customers view the system as a very important part of the program because it meets the simulator loading requirements and has a high reliability.

Acknowledgment

I wish to acknowledge and thank members of the I/O hardware development consisting of myself, Gary Ojala (DBT Lead), Pat McCuen (Hardware design lead), Ted Nasi, Jerry Friedrich, Jim Mueller, Carol Wolfe, Frank E.Smith, Greg Mars and (Art Safarli of the FSL Central Design group).

I wish to thank the members of the Design Build Team (DBT) consisting of many Boeing FSL groups.

I wish to acknowledge and thank the FSL customer Don Brooks, and J.L. Abbott for their support through out the Boeing 777 program .

I wish to acknowledge my supervisor Elias Beyrouty for his suggestions and for his help on the project.

I wish to acknowledge my wife, Mary for her patience, and encouragement during this project.

AIAA-94-3411-CP

INCREASING THE FLEXIBILITY AND REDUCING THE COST OF FLIGHT SIMULATORS

E. Theunissen and T. Lamerigts***
Faculty of Electrical Engineering
Delft University of Technology
Mekelweg 4
P.O. Box 5031
2600 GA Delft, The Netherlands

Abstract

At Delft University of Technology a new Flight Simulator, called the Simona Research Simulator is being developed. Since the main goal of this simulator is research, it is important to be able to configure systems such as control loading, motion, the Electronic Flight Instrument System, the Flight Management System and sound-generation with a high degree of flexibility. This paper presents the approach which was used for the development of these systems. Our goal can be summarized as 'To obtain a highly flexible system, and minimize overall system cost'. The approach is based on an efficient translation of requirements into design criteria, an optimized hardware/software design, and advanced software development techniques.

Introduction

The Faculty of Aerospace Engineering at Delft University of Technology operates a 3 Degree-Of-Freedom (DOF) moving-base flight simulator for research and educational purposes. In 1995 this simulator will be replaced. The new flight simulator, from now on referred to as the SRS which is an acronym for Simona Research Simulator, is being developed in a joint operation between the Faculties of Aerospace Engineering, Electrical Engineering, and Mechanical & Maritime Engineering. It is apparent that to succeed, the cooperation requires the seamless fusion of domain specific knowledge from the encompassing disciplines.

It is important to recognize that due to the higher frequency of changes in requirements the lifecycle of a research simulator differs from that of a training simulator, which necessitates a higher degree of flexibility for the former one. To achieve the goal of obtaining a highly flexible system and minimize total system cost, attention must be paid to the factors which determine flexibility and the factors influencing system cost during the entire lifecycle of the system. This lifecycle begins with system development which encompasses the specification of the requirements, the specification of the system, the development and/or acquisition of the required components, and system integration. After successful system integration and testing, the operational status is achieved. During the operational lifecycle of the system, requirements with respect to operatability, (re)configurability, serviceability, and continuity of service must be met. Also, maintenance and upgrading is necessary to enhance and update the system in order to meet changing and new requirements. The lifecycle ends with the decommissioning of the system.

Since flexibility and initial cost require a trade-off, total cost must be divided into initial cost, and the recurring system cost. As more flexibility requires more up-front investments, development cost will increase with increasing requirements on flexibility. However, a well designed system should be able to pass the break-even point through a reduction in maintenance cost. To get a better picture of the initial cost, the following subsections make a distinction between development cost, system acquisition cost, and cost resulting from required patches or partial redesign. The recurring system cost are divided into operational and maintenance cost. After an identification of the possibilities to reduce cost, these possibilities are discussed in the context of the translation of requirements into design criteria, hardware design and software design.

The remainder of this paper is organized as follows: In the section about hardware, the development of the network and the display system for the Electronic Flight Instrument System (EFIS) are discussed. In the section about software, much attention is paid to the existing software paradigms, and how they fit into our requirements. Furthermore, it is discussed how one of these paradigms has successfully been applied for the development of display software.

Development cost

In this project, many participants from different disciplines are interacting in the design process of a complex system which is characterized by highly interactive hardware and software. In such a process, it is of crucial importance to integrate software requirements with hardware design synthesis. Due to the rapid increases in performance of computer hardware the situation is very complicated. Plant[18] refers to the effect of the rapidity of technology changes in the area of information technology as the "ΔT effect". He states that 'unless management and developers control their software processes through an understanding of the impact that hardware, software, practical and theoretical developments have upon it, these technology changes may have a serious and detrimental effect upon the developers software creation

*PhD candidate, Electrical Engineering

**MSc. Electrical Engineering

process'. To reduce development time and cost it is important to recognize the iterative nature of the design process and the fact that it is impossible to proceed in a serial manner. Thus a concurrent engineering approach is required. For such an approach to succeed, good communication between the different disciplines is essential. Communication problems can result in conflicting or misinterpreted information, which reduces productivity and may require patches or a partial redesign of the final product. As a result, the required fusion of knowledge is only obtained at very high cost. Since the anticipated system is meant for research purposes, the problem becomes even more severe as certain phases which are normally only encountered during system design now will re-occur during the operational lifecycle and can be considered as part of the maintenance process. For efficient communication a framework is required which allows the participants to translate their requirements with the supporting rationale into the specific domains of other engineering professionals and obtain understandable feedback. The impact of the supporting rationale should not be underestimated since very often improvement is only possible if the reason for change is understood and accepted. In this paper an example framework is presented which during the development of the EFIS aided in the systematic translation of high level design requirements into low level hardware selection criteria.

To assure that during the operational lifecycle changes in the functional specification do not require major system modifications, the need for flexibility becomes paramount. This justifies an increased up-front investment in flexibility during system development. Flexibility is for a large part determined by software, and therefore the typical software lifecycle for the systems should be identified, and a software development paradigm must be selected which best fulfils the requirements.

System acquisition cost

A large part of a flight simulator is determined by computer hardware and software. Examples are systems for functions such as control loading, EFIS and Flight Management System (FMS) simulation, motion control, and sound-generation. A wide spectrum of computer hardware is available for these purposes. Very often even a Personal Computer (PC) offers the required computing performance. The problem is to select the hardware with the best price/performance ratio over the entire simulator lifecycle. This requires taking into account factors such as scaleability and upgradeability, so in case an increase in performance is required, only the components responsible for the bottlenecks in performance need to be updated. Often software cost exceed those of the hardware. Due to the broad user community of PCs, software cost are relatively low and quality is relatively high compared to most workstations.

System operation cost

Flight simulators, whether used for training or for research, are very expensive. Therefore, the total time the system is down and cannot be used for the purpose it was designed for must be kept to an absolute minimum. Maintenance and repair activities may cause the system to be unavailable for a certain amount of time. The system must require minimum maintenance for continuous operation. A high degree of modularity and commonality among the different subsystems contributes to better maintainability.

Since maintenance activities can be planned, the Continuity Of Service (COS) provided by the flight simulator is determined by malfunctions which require the system to be shut down. The total time the system is down due to a defect hardware component must be kept to an absolute minimum. System functions should be divided over the hardware in such a way that failure of a certain component causes minimal disruption to the total system operation.

A combination of four different approaches can be used to achieve the desired COS. The first approach is to design for redundancy. In case of a single failure, no system degradation results. The second approach is to operate a system in a degraded configuration. The possibility to do this depends on the reconfigurability. The third approach is to perform an immediate service in order to keep the system operational. The fourth option is to replace the defect components. This is determined by the replaceability, which in turn is determined by the modularity of the system and the commonality of the subsystems. A factor which affects both maintenance and repair is the serviceability, which indicates the effort required for a certain service to be performed on a system.

Maintenance/upgrading cost

Since functional requirements for a research simulator will change much more frequently than those of a training simulator, different approaches are required for efficient system maintenance. A research simulator should be able to keep up with advances in technology during its entire life cycle. When more computing performance becomes available, there will be a demand for this performance. For example, the mathematical models used to describe and simulate the behaviour of a dynamic system require a trade-off between the quality of the match with the actual system and the required computing performance. Consequently, systems responsible for performance bottlenecks should be easy to upgrade as faster ones become available. Another option is to provide a scaleable solution, in which more processing power can be added. Software should be designed in such a way that when upgrading hardware, no major re-engineering or programming is required. In other words, the software lifecycle should be extended beyond the hardware lifecycle. As a result it is necessary to design with the objective of upgradeability and scaleability in mind. Otherwise, when requirements dictate an increase in performance and the cost of upgrading become too high as a result of the required system integration, the upgrade has to be abandoned. For hardware, a major factor influencing upgradeability is modularity.

Simulator hardware and software should be developed with the reality of change in mind. This implies that the system must provide adequate flexibility to the user and should be easy to reconfigure. This has implications for the system functions implemented in hardware and the functions implemented in software. It is apparent that the ability to change the configuration through software increases the reconfigurability. To minimize software maintenance cost, appropriate software development techniques must be employed. Modularity and commonality among subsystems are a major factor in satisfying requirements with respect to maintainability, serviceability, upgradeability, (re)configurability and replaceability. Redundancy can be used to increase the continuity of service.

Translating requirements into design criteria

A successful design requires the optimization of design parameters to best fulfil the system requirements within the given constraints. The system requirements, in most cases defined by the end-user, have to be translated into hardware and software design criteria. When translating requirements to a more design specific level, it often appears that certain requirements are contradictory or require a trade-off. As a result of missing implementation specific domain knowledge, the information about relations between different requirements which is necessary to make certain decisions is often invisible at the end-user specification level. Thus, to identify and compare possible implementations and provide feedback to the end-user for further requirements specification and fine-tuning, communication between the system designer and the end-user is required. As a result of this communication, requirements are better understood and can be incrementally improved and clarified. Often new options which neither the end-user nor the system designer alone could have foreseen become apparent. For efficient communication a framework is necessary which allows the translation of high level end-user requirements into selection and design criteria. Based on such criteria, systems can be proposed which fulfil the end-user requirements.

To solve the communication problem, the different domains involved must be identified, domain boundaries must be defined, and the relations between the different domains must be described. Also, technological limitations must be indicated. During the design process it is important that the rationale for the choices which are being made are understood by those involved. In this way, the end-user is better able to see the relationships underlying his specification, understand the reasons for limitations and required trade-offs, and predict the effect of changes in his requirements. It also allows him to better understand the problems the system designer is faced with. The following example indicates how certain high level requirements for the EFIS display system are translated into design criteria.

For the EFIS display system, certain requirements which are determined by the environment and the perception criteria must be met by selecting the appropriate display device, e.g. a Cathode Ray Tube (CRT), or Liquid Crystal Display (LCD). Requirements with respect to e.g. resolution, number of colors, and speed determine the graphics hardware and software. The constraints are determined by the environment in which the equipment is to be integrated and operated. Requirements following from the tasks to be performed include the amount of information which must be presented on a display and the update rate of information. Requirements resulting from perception criteria address e.g. angular resolution, refresh rate, and the number of available colors. A graphics system is characterized by parameters indicating the available display memory, horizontal and vertical scanning frequencies, resolution of the Digital to Analog Converters (DACs), and computing performance (often expressed in MIPS, MFLOPS, Gouraud shaded polygons per second, etcetera). Parameters specifying a display device depend on the type of device. For a CRT, display size, dot pitch, horizontal and vertical scanning frequencies, contrast, and luminance are important parameters. Environmental requirements and constraints determine total system size and weight, maximum power consumption, heat generated, and shocks and vibrations the system will be subjected to. In this example, the translation from end-user domain specific requirements to design specifications covers 4 levels. Level 1 is the end-user specification level and level 4 the hardware level. At this level the design parameters for the system are specified. For the display system these include memory, scanning frequencies, resolution of DACs and MIPS. Level 2 contains the specification of the devices which are needed, e.g. display device and graphics engine. Level 3 contains the high level design parameters for these systems. For a graphics system these would include e.g. resolution, colors, and speed. The relation between level 3 and level 4 is indicated by expressing the dependencies of the level 3 parameters as a function of level 3 and level 4 parameters. The reason for allowing the specification of level 3 parameters as a function of other level 3 parameters is allowed to indicate relations which directly influence each other and as a result may require a trade-off.

Hardware

To reduce hardware maintenance cost a modular hardware architecture is used, and commonality is emphasized. To reduce initial and maintenance cost the use of dedicated systems is avoided and of-the-shelf equipment is used whenever possible. As a result of the increase in performance of PCs, it has become possible to use of-the-shelf PC equipment instead of dedicated hardware or workstations. This has a number of advantages.

First of all, since systems based on dedicated hardware and workstations are often quite expensive, hardware cost are reduced. This is not only the case for the initial cost, but also for maintenance.

Second, due to the large market share of these systems, development tools are much cheaper.

Third, a compatible development environment is created since the same type of equipment is used to perform a variety of functions, such as networking, voice generation, voice recognition, EFIS and FMS simulation, and control loading.

Fourth, due to the higher commonality connectivity among different systems is increased.

Fifth, flexibility is increased relative to systems based on dedicated hardware because PC based systems enjoy a variety of very advanced high quality development and debugging tools.

Finally, due to the resulting commonality among the systems, maintainability is improved.

In our design we emphasize the use of PC based systems, and as a result much of the hardware of different systems is interchangeable. The implication of this approach on system cost should not be underestimated. PC based systems in the SRS include the EFIS, the FMS, the control loading system and the motion control system.

Software

The software for a flight simulator can be divided into the following three categories: simulation of the aircraft model, motion control software, and Man-Machine Interface (MMI) software. To assure that during the operational lifecycle changes in the functional specification do not require major system modifications, the need for flexibility becomes paramount.For efficient software development it is important to recognize the difference between the typical software lifecycle of a training simulator and a research simulator. The lifecycle of the latter one is characterized by a much higher frequency of changes in requirements and hardware environment. Since a research simulator should be able to keep up with advances in technology during its entire lifecycle, systems responsible for performance bottlenecks should be easy to upgrade as faster ones become available. As a result, it is necessary to design with the objective of upgradeability in mind. For the software this means that methods should be used which allow easy rehosting of the existing software to new target hardware. Only in this way it is possible to extend the software lifecycle beyond the hardware lifecycle. Thus, software should be developed with the reality of change in mind, both with respect to changing user requirements and changing target hardware. To satisfy the first condition, software should be flexible with respect to changing requirements, support end-user modifiability of the functionality, and have a high degree of maintainability. To satisfy the latter one, software should be developed for hardware independency. Besides these requirements, software has to satisfy certain constraints with respect to reliability and performance.

When software is easy to rehost and requirements are characterized by the wish for more performance, there is no software problem. However, the complexity of requirements determines software complexity and an often encountered problem with existing software development paradigms is their inability to efficiently scale up to large and complex projects. Therefore, the flaws responsible for this problem must be identified and eliminated.

One of the problems which hampers the efficient design of systems is the fact that in many cases end-users are initially unable to specify complete requirements. This is also true for software. Furthermore, requirements may be wrong in some details, or subject to change during the lifecycle of the software. Consequently, software development should not be considered as a serial process. When design principles adhere to a serial process model this can result in several problems. First of all, it is possible that the software does not meet the requirements, resulting in costly patches or partial re-design. Second, if the designer is unaware which requirements are subject to change during the lifecycle of the software he may not have included enough flexibility to provide for an efficient implementation. Finally, in case the designer builds in too much flexibility, the system may become too complex, slow or expensive, which is all equally undesirable. These problems are caused by the lack of feedback to the end-user in the early design stages. Various approaches to this problem have been proposed[2,6,15]. To select the one which best satisfies our requirements, a better understanding of the typical software lifecycle in a research environment is needed.

Software lifecycle models

Various models have been proposed to describe the software lifecycle. Davis[9] presents a strategy for comparing alternative software development lifecycle models. Most of the different life cycle models are either of the waterfall[5] or of the prototyping model type. The latter one can be divided into rapid prototyping, where a 'throw away' prototype is generated for initial evaluation purposes and evolutionary prototyping, where the prototype is successively refined into the final product. In this way lifecycle cost are reduced since changes made at the early design stages are much cheaper than changes to a finished product. A basic difference between the waterfall model and the prototyping model is the feedback from later to earlier phases. This feedback provides information on how decisions made at one stage are reflected later in the design, and therefore can be used to incrementally improve the earlier phases. This requires more communication between the end-user, the system designer and the programmer and can consume considerable time. For efficient communication between the software designer and the end-user, a common frame of reference is required. To achieve such a common frame of reference, domain specific, informal, and often ambiguous methods are used. The designer translates these requirements into a software design. Possible ambiguities are resolved through communication with the end-user.

The spiral model[6] which involves a repetition of specify-design-code-validate cycles, is representative for the approach on which an easily specified prototype is extended on each cycle, and can be classified as an evolutionary prototyping approach.

Based on the concept that 'the larger the project is, the more there is to gain from up front investment' Levy[15] proposes "metaprogramming" to reduce cost by reducing total system development time and increasing quality. Metaprogramming is an approach in which first a 'throw away' prototype is generated based on a certain set of requirements which remain fixed. Once the prototype is completed and the form of the final system is fixed, a set of application generators are built which construct the real system directly from the specifications. Further system modification is done by changing the input to the generators.

In Balzer's automation based paradigm[2], the software lifecycle is split into a repetitive prototyping cycle containing the specification-design-evaluation phases, followed by a coding phase. An increase in productivity relative to the spiral model can be obtained since the coding and validation process has been taken out of the prototyping loop. His new knowledge based software paradigm is characterized by the fact that all software lifecycle activities are machine mediated and supported, or in other words 'the machine is in the loop'. In this new software paradigm, the actual implementation is automatically derived from a formal specification.

Harel's[12] "vanilla" approach to modelling discusses executable specifications as a means to test the model before actual implementation. This execution requires an interpreter which can run the specification to simulate the dynamic operation, thus allowing extensive prototyping and simulation of the system in the early design phases. A further advantage of model execution is that the possibility increases that unintended patterns of behaviour are discovered before system implementation. Balzer[2] indicates that in certain situations the interpreted specification satisfies all the end-user needs, and code generation for implementation becomes unnecessary. An advantage of interpretation compared to code generation is the high flexibility and the fact that no compilation is necessary after changes have been made to the specification. The important advantage of code generation is the increased performance and the reduced size of the resulting software, which can become even more important when code has to be ported to other target environments.

Impact for a research simulator

For all three previously mentioned categories in flight simulator software, possibilities for high level approaches to software design exist. In the aerospace community, the concept of automatic code generation is increasingly being accepted and used as a means to increase productivity. Code generators can reduce cost and development time and increase reliability and performance. As a result, the developer can focus on designing functionality and evaluating performance instead of writing and debugging code. An additional advantage of automatic code generation is that to the extent that the formal manipulations which are used to generate code from a specification can be proved correct, the validation paradigm will be radically altered[11]. Rather than testing the resulting implementation against the specification, the validity of the implementation arises from the process by which it was developed. At the Navigation, Control & Aeronautics Division (NCAD) at NASA-JSC, commercially available software products are used to develop Guidance, Navigation and Control (GN&C) algorithms in block diagram form and then automatically generate source code from these diagrams. An increase in productivity of 185% is reported[7] over the traditional methods used for developing flight software.

MMI software comprises the software for control loading systems, the EFIS, the FMS, and the Electronic Library System (ELS). The software complexity of the MMI systems is for a large part determined by the User Interface (UI), which relies heavily on display software. In the rest of this section an approach to achieve a high level of flexibility, required for a rapid turnaround time in display format research, is described. Compared to most existing rapid prototyping tools for display format development, this approach is characterized by its ability to smoothly progress beyond the prototyping stage for simulator evaluation and in-flight testing.

Research into display formats is a typical iterative process consisting of a prototyping phase and multiple evaluation/modification cycles. Beadon[3] distinguishes between four different phases in display format development: static formats, animation, high fidelity simulation, and testing within the vehicle. This requires the development and maintenance of software in an environment which is characterized by a high frequency of changes in the specifications. Furthermore, part of the software has to run on dedicated systems, which requires a high level of domain specific expertise. As the system is being used in a research environment, programmers (in our case often students) frequently have to modify part of the display software to change existing features or to implement new ones in order to meet certain requirements. The process described above sums up some of the typical problems encountered with software:

1. Domain specific expertise required.
2. Minor changes in the specification often take quite a programming effort for implementation in software.
3. Due to the always changing programmers software maintenance is difficult.

These problems decrease flexibility. A software development system should allow the end-user to specify his requirements, support the software designer with the translation of these requirements into a design, and support the programmer with the translation of the design into code.

Due to the different phases in display format development still multiple implementation and evaluation phases are necessary, increasing turnaround time. The automatic programming paradigm[2,11] presents a solution to this problem, and the rest of the software discussion in this paper will focus on this approach. The automatic programming paradigm is based on the principle of 'the machine being in the loop' which requires that the acquisition of requirements, software design and software construction are all machine mediated. To be able to apply the automatic programming paradigm, a software development system is required. The ideal situation is best described by the cocktail party description of automatic programming[19]: 'The end-user writes a brief requirement for what is wanted, and the programming system will produce an efficient program satisfying the requirement'. Since such a system should be considered a myth, a more realistic approach is required, which is presented in the subsection "Possible approaches".

Automatic programming

In the April 1958 issue of Communications of the ACM, a chart lists the availability of "automatic programming systems". Most of these systems, we now refer to as assemblers, and some of them as compilers. To avoid confusion, the following distinction is made between an assembler, a compiler and a code generator: An assembler translates a symbolic representation for a numeric language into this language, a compiler translates a problem oriented language into either a numerical machine language or a symbolic representation for one. A code generator is a program which translates a specification containing the required functionality of a program into a problem oriented language. Thus, a code generator translates information from the specification level to the implementation level. The first compilers for translating source code to assembler were developed in the fifties. Code generators for producing source code from a specification first emerged in the seventies.

The difference between a specification method and a programming language is that a specification only contains the information about the required functionality, whereas a program contains a method to achieve this functionality. As a result, automatic code generation requires the availability of a specification method to define the desired functionality. The specification method used can vary from generic to domain specific. Generic approaches are characterized by formal techniques. In contrast, domain specific approaches relate knowledge about the application to the desired software at the requirements, design or implementation phase.

Automatic code generation can be performed from a specification of the software design, or from a specification of the application design[23]. Application design is an abstraction level higher than and takes place before software design. Software design translates the required functionality of the application to the software design level, which takes place on a higher abstraction level than code generation. McCartney[17] describes this high level design as 'the process of going from a functional specification of what a system does to an operational one. The output describes the structure of the solution in terms of algorithm structure, control, and data flow among modules whose description is independent of a particular implementation language'. The high level design is followed by code generation, which translates the operational specifications into code in some language. Software design is determined by the characteristics of the execution environment, whereas code generation is determined by the characteristics of the target language.

Possible approaches

Rich[19] identifies three approaches to automatic programming, the 'bottom up approach', the 'narrow domain approach' and the 'assistant approach'. The first one is characterized by the use of very high level languages. The second approach focuses on the concept in which the end-user specifies his requirements in a formal way, and uses an automatic code generator to translate these requirements into software. The third one, the assistant approach focuses on a system which supports the software designer with the design and the programmer with the implementation of the software, thus sacrificing the principle of full automation. Productivity is increased through the use of so-called assistant tools.

The second myth of Rich[19] says 'End-user-oriented, general purpose, fully automatic programming is possible'. In his description of reality, currently only the narrow domain approach seems feasible to construct a fully automatic program generator which directly communicates with the end-user.

Lowry[18] indicates that automatic code generation requires trade-offs between several dimensions. He identifies the following five:

1. The distance spanned between the specification level and the implementation level.
2. The breadth of the domain covered at the specification level.
3. The efficiency of the implemented code.
4. The efficiency and degree of automation of the translation process.
5. The correctness of the implemented code.

Another approach to increase productivity is to stimulate reusability. However, software is only reusable if it was specifically designed for this purpose. The problem with software reusability follows from the fact that current approaches focus on reusability at the implementation level by means of providing libraries containing a collection of specific functions. The fact that the function is always accompanied with all kinds of implementation specific constraints often renders such libraries useless. Angus[1] states that many of the problems which can occur when reusing software components can be traced to ill-defined interfaces describing the existing application. He presents some guidelines how these problems can be prevented in the future. Fisher[10] describes a method where reusability is provided at the specification level. With this approach an existing specification can be tailored to specific needs, after which the implementation is generated. This is comparable to the reuse of display format specifications in the Delphins Display Design System (D^3S)[21].

Identification of pitfalls

As indicated by Balzer[2], one of the fundamental flaws in the existing software paradigm is the fact that maintenance is performed at the implementation level rather than at the specification level. Of course this is still possible with automatic code generation and to prevent this from happening, the reasons for modifying the source code must be uncovered and eliminated. The problem is typical for code generators which produce source code from a specification, but hardly occurs with compilers. With the current software paradigm everybody will agree that when a modification must be made to a program and both the source code and the object code is available, the modification is made at the source code level. Why? Because it is the easiest thing to do. If we are able to create the same situation with automatic code generation, nobody will try to modify the generated code, and the problem mentioned previously will disappear.

Several reasons for modifying the display software can be identified. These reasons have been divided into the following three categories:

1. Allow the software to communicate with other modules.
2. Make quick changes to the display format.
3. Increase speed by optimizing the source code.

Once a change has been made at the implementation level, the user is confronted with the dilemma that when new changes are required, all changes which were made at the implementation level must be performed again in case the new changes are made at the specification level. This will discourage the use of the code generator for small changes in the specification of the display format. As a result, these changes are made in the source code, and the consistency between the specification produced with the development system and the display program is lost. This approach is sure to backfire in the long term as it is a very tedious task to keep track of all changes so they can be implemented in the next version of the automatically generated code.

Cleaveland[8] indicates that since a code generator is frequently part of a larger system, interfacing the automatically generated code and the rest of the system is very important. The second of his seven steps[8] to building a generator therefore requires the definition of domain boundaries, and an answer to the question of what and where the interfaces should be.

Another method to discourage the user from modifying the source code is to design the system in such a way that the representation of the objects is not in the source code, but loaded at the start of the program. Following the idea of 'what is hard to read is hard to understand and even harder to modify', an option to discourage people from modifying automatically generated source code, proposed by Knuth[4], is to generate hard-to-read programs.

The wish to optimize speed is another motivation for changing the source code. The graphics hardware and software determine the update rate of the displayed information, given the update rate of the information itself is high enough. The cost of the graphics hardware grows with increasing requirements on the information update rate. To produce smoothly animated pictures, a minimum update rate of approximately 20 Hz is required. Update rates can be increased through clever programming of the graphics hardware. An experienced programmer will know what methods to use to obtain the highest performance. In contrast, a code generator does not contain this experience. In order for a code generator to produce optimized code, a rule base, capable of applying the methods which an experienced would use, is necessary. The motivation to modify the code to increase speed will disappear when the efficiency of the generated source code exceeds a certain threshold.

Besides these pitfalls, another aspect is very important. The system itself should be user friendly, require limited training for basic usage, and couple a gradual degree of desired complexity to a gradual increase in required expertise. In this way, a low entry level is enabled, a broad user community can be realized, and a better control of training cost is possible. If the system is only understood by a few experts, and the threshold of becoming an expert is too high, non-experts will still fall back on manual programming.

Results

At Delft University of Technology, PC based hardware is being used for a variety of functions in the future Simona Research Simulator. Examples are the EFIS, the FMS, the control loading system, the motion control system, and the sound generation system. Advanced software development methods are used for a majority of these systems. The EFIS, the network, and the D^{3}S software development system are already being used with the current flight simulator. In this Section, these systems will be discussed in more detail. Besides these examples, similar approaches are being used for the development of the navigation system simulation and the motion control system. For the latter one, commercially available software packages are used to support the designer with the specification of the desired functionality, after which code is generated for the target system, consisting of a number of PCs with Digital Signal Processor (DSP) boards.

The EFIS

The SRS EFIS comprises six display stations and one master station. Each display system provides a basic interactive capability by means of a touch screen. The functions of the EFIS master station include control, monitoring, and data recording.

System configuration

Several configuration options for the hardware are possible. When the display system hardware would be installed off platform, this requires cabling for several input devices (e.g. keyboard, touchscreen) running from the simulator to the display hardware. Also, about twenty coaxial cables would run from the moving platform to the lab. Another option is to install all hardware on the platform. However, in this case the equipment will be subjected to a wide spectrum of vibrations. This is especially troublesome for mechanical parts such as floppy or hard-disk drives. Since these devices are only needed to load the necessary display software into the memory of the display computer, and it is possible to boot the system from the network, the on-board display system can be made completely diskless and keyboardless. Since the resulting display hardware can be built very compact, it was decided that it will be installed on the simulator platform. All display systems are connected to the host through a network.

Hardware

The EFIS should be able to simulate existing EFIS display formats. Furthermore, simulation of advanced concepts such as three-dimensional display formats and Enhanced Vision Systems (EVSs) should be possible. PC graphics engines range from low-end single processor systems which provide high resolution images at a sufficient update rate to high-end multi-processor graphics engines, providing features such as hardware polygon rendering, Z-buffering and Gouraud shading. Even the high-end PC graphics engines which are able to generate complex 16 million color pictures with a resolution of 1280 by 1024 pixels at an update rate of 30 Hz are available at a fraction of the cost of graphics workstations with comparable performance. Since not all six display stations are required to present EVS images, the range of available graphics engines allows a scaleable solution.

Upgradeability

Upgradeability is determined by two factors: hardware compatibility and software compatibility. Due to the use of of-the-shelf PC systems and the large installed base of this type of equipment, hardware compatibility with future systems is very high. When upgrading to a new host for the graphics engine, no modifications to the display software will be required. When upgrading to a new graphics engine, two options are possible:

1. The new graphics engine is compatible with the current one. In this case, no modifications to the software are required.
2. The new graphics engine adheres to a different standard. This requires that part of the existing software must be modified. However, due to the approach used[21], only the program which executes the display list must be rewritten. All other programs do not require any modifications.

Realization

Development work on the display system hardware and software started in 1990 in the context of the Delft Program for Hybridized Instrumentation and Navigation Systems (DELPHINS), and a prototype system has been installed in the current flight simulator. This system comprises a 486 based host, supported by a TMS34020 graphics processor.

The network of the SRS

All computer systems in the SRS are connected to the host by means of a network. For the development of the network, the simulator software is regarded as a distributed set of tasks which execute periodically. It is assumed that sporadic and aperiodic tasks can be transformed into periodic ones. A periodic task can be characterized by four parameters: the task period P, the task's (maximum) computation time per period C ($C \leq P$), the task's deadline with respect to the start of a period D ($C \leq D \leq P$), and the task phasing with respect to some reference time I. In other words, the first execution must end at time I + D, the second at I + P + D and so on.
Some or all of the tasks may be communicating tasks, which means they transfer data (via a network) to other tasks on every invocation. The time a task spends communicating is part of its computation time C. Because the task is periodic, it generates messages periodically. Each message can be viewed as a communication task, so communication can also be viewed as a distributed set of tasks with parameters P, C, D and I that follow from the parameters of parent task. For out purpose it is assumed that a communicating task sends exactly one message per invocation.

Requirements

The network should be able to operate in a non-deterministic initialization/shutdown and maintenance mode, and in a deterministic real-time simulation mode. In the non-deterministic mode the network must offer remote-booting of nodes. In deterministic mode the network should be able to always make all (communication) tasks meet their deadlines. The remainder of this section focuses on the deterministic mode. As previously indicated, for reasons of initial and maintenance cost, the network hardware should consist of of-the-shelf available components.

Selection of hardware and protocol

In a system with a set of n periodic communication tasks the aim is to select network hardware and a (software) protocol such that all tasks always reach their deadline. This means finding suitable values for P_i, C_i, D_i and I_i ($1 < i < n$). P_i and D_i are mainly determined by the simulated environment and the perception criteria. C_i is a design parameter that follows from two communication properties: maximum channel access time t_a and transmission rate R. R is a hardware property, while t_a is, in general, determined by both hardware and software. I_i can be selected freely as the simulation model permits.
If I_i is assumed to be arbitrary, Rate Monotonic (RM) analysis can be applied to verify whether all deadlines are met. This approach requires a deterministic network access protocol, typically implemented in hardware, e.g. a token passing protocol or Time Division Multiple Access (TDMA)[13,22].
Better predictability and testability can be achieved by using a network access protocol that allows the transmission instants of all messages to be determined and analyzed a priori. For the SRS, we have chosen P to be equal for all communicating tasks. P is small enough to satisfy all simulation and perception criteria. In that case, assuming a network that can handle the total load of n communicating tasks, we can select I_i ($1 < i < n$) such that all deadlines are met. Conceptually, this is equivalent to TDMA. Appropriate selection of I_i automatically avoids multiple (simultaneous) channel access, so (hardware-based) collision avoidance logic is no longer necessary. In fact, some collision avoidance techniques, like token passing, in this case needlessly increase t_a and hence might cause sub-optimal utilization when all deadlines are met.

The task phasing in the system must be maintained across physically separated nodes, so the nodes must be synchronized. To avoid clock synchronization problems our protocol uses only one central clock in the host computer. The beginning of each TDMA cycle is triggered by this clock, and the task phasing is chosen such that one communicating task running on the host computer always transmits at the beginning of the cycle. Note that the host may have multiple communicating tasks, which means it can transmit multiple times per cycle. Because the task phasing is known a priori to all nodes, the other nodes can determine when it is their turn by just listening to the network, so the network traffic is synchronized via the network itself.

Error detection can be performed by having several or all nodes listen to other node's transmissions. Because it is known a priori who shouid transmit at a certain time instant, nodes can detect omission errors or out-of-order transmissions, and signal other nodes there is an error. Performing error detection costs processing time so the number of nodes doing it can be selected according to available processing power and desired coverage.

Since it offers adequate performance and is very well-suited for our protocol we have selected Ethernet. Ethernet has R = 10 Mbps and, using our protocol which avoids collisions, t_a = 0. The protocol is fully implemented in software[14]. The non-deterministic services are performed by commercially available software.

Scaleability

While there is still network bandwidth left, a node may be added by simply attaching it to the Ethernet cable, choosing an appropriate time phasing for it, and rebuild the protocol software on all nodes with the new parameters. If network bandwidth is fully used, the host can be expanded with an additional Ethernet adapter, and nodes can be added to the new branch. Multiple Ethernet branches are independent and can be operated (nearly) concurrently.

Note that it is possible to automate the generation of a time phasing schedule and the protocol software implementing it.

Realization

Development of the network, the Baresim System Interconnection (BSI) started in 1991. A prototype system was demonstrated in 1992[14], and a fully operational configuration driving several display systems became operational in 1993. The network is also used in our laboratory aircraft, a Cessna Citation II. In this aircraft, each chair is equipped with an interactive display system, consisting of a high resolution color LCD and a 386SX based Single Board Computer (SBC). Interaction with the system is possible by means of a touch screen. A host computer collects aircraft information through a variety of interfaces, and transmits this information to the display systems.

The DELPHINS Display Design System

The development of display software for the research simulator and the laboratory aircraft is based on the automatic programming paradigm[2,11]. A system for the development and evaluation of display software, D^3S, has been realized in 1991[21]. Since then experience has been gathered with D^3S and the system is now being used for the development of display software in our laboratory, in our research flight simulator, and in our laboratory aircraft. We succeeded in combining the size and speed advantages of code generation with the flexibility of specification interpretation, and in our goal to eliminate the maintenance of software at the implementation level. Our experience with the development of the EFIS and the software development system D^3S shows that an upfront investment in flexibility and upgradeability pays off very fast due to the high frequency of the recurring maintenance cost during the operational lifecycle of a research flight simulator. Conventional as well as various advanced display formats for research purposes, e.g. Tunnel-in-the-Sky[20], have been developed with this system.

The D^3S philosophy is to provide computer support at the requirements acquisition phase, the UI design phase, the software design phase, the code development phase, the software integration phase, and during maintenance and rehosting. All transitions between these phases are computer mediated, keeping all participants in the same loop through computer support.

Our approach aims for end-user oriented automatic code generation. In this approach, experienced software designers and programmers are still required for the system integration of the display software. To support them, tools which translate the end-user oriented specification into a specification more suitable for system integration are needed.

By developing a domain model which makes it easier for the user to implement changes in the specification than in the source code, the motivation to make changes at the implementation level disappears.

Specification

For our purpose, a specification method, the so-called Display Definition Format (DDF), which is very close to the user has been developed. A large distance is spanned between the specification and the implementation level, and the requirements can be specified in a narrow domain.

To eliminate the necessity of making changes for communication purposes, a standard interface towards the display software has been defined in D^3S[21]. The high level part of the interface is connected to the inputs of the display program, and the low level part allows direct access to the buffers containing the object descriptions.

To speed up the process of acquiring a workable initial specification, the UI designer is supported with tools which allow him to specify an initial design, evaluate its behaviour and refine it through the feedback obtained this way.

Implementation

The actual implementation is derived from the specification via a series of formal manipulations, often referred to as transformations. Vertical transformations are used to reduce the level of abstraction, and proceed from the specification level to the implementation level, and horizontal transformations are used to optimize the code. An advanced code generation facility allows the automatic construction of a real executable prototype for further evaluation.

The degree of automation in the translation process is very high. In fact, no interaction with the user is required.

The efficiency of the implemented code is very high, and situations in which it is useful to modify the generated code to increase display update rates are rarely possible.

The approach to code generation used in D^{3}S combines most of the flexibility from executable specifications with the advantages in speed and size of code generation. The first specification-evaluation cycle comprises the process of acquiring a workable initial specification. Once this specification is complete, the implementation is generated. This rapid prototyping capability reduces the effort required for initial concept evaluation. The second specification-evaluation cycle represents simulator or in-flight evaluation. Since the specification is loaded at the start of the program, changes to the display format can be implemented in the specification without having to regenerate the software. Thus, code generation has been taken out of the second specification-evaluation cycle, which reduces the turnaround time for simulator and in-flight evaluation. Furthermore, it prevents the user from making changes to the display format at the implementation level, since the information simply is not in the source code.

Software integration

Integration with the flight simulator or the laboratory aircraft requires the assistance from software specialists. To support these specialists, various tools can be used. To obtain an overview of the software design and the communication with the other systems, the specification can be translated into diagrammatic charts. To support the programmer with the integration of the software, source files containing the exact specification of the interface of the display software are automatically generated. This is especially useful since this information is both required for the transmitting modules and their receiving counterparts.

Maintenance and Reuse

A variety of tools can be used for display software maintenance at the specification level. Since the specification is easy to understand, in most cases a text editor suffices. For a modification of the layout, a graphics editor is more appropriate.

Since reusability is provided at the specification level, where the functionality is not yet influenced by all kind of implementation specific details, this method also encourages reuse of existing display formats.

Upgradeability and rehosting

To rehost the display software to the selected target system, code generators for the various target systems (PC, TMS34020, IRIS) have been developed. By designing for hardware independency and separating hardware independent and hardware dependent mechanisms, the effort required for upgrading to newer systems has been minimized, while the ability to optimize for the target hardware remains. In this way, the software lifecycle can be extended beyond the hardware lifecycle.

Conclusion

It is important to recognize that due to the higher frequency of changes in requirements the lifecycle of a research simulator differs from that of a training simulator, which necessitates a higher degree of flexibility for the former one.

The establishment of a framework to translate user requirements into hardware and software design requirements, and the integration of software requirements with hardware design synthesis allows an efficient iterative design process. In this way, it is possible to develop and optimize a system for the specific application and avoid the situation where over-dimensioned and thus too expensive systems are acquired.

System development and acquisition cost can further be reduced by using of-the-shelf equipment and components instead of buying or developing dedicated hardware or using workstations. The development of the EFIS and the network described in this paper are examples of this approach.

With respect to software development it is important to recognize the difference between the typical software lifecycle of a training and a research simulator. Furthermore, it is important to recognize the need to go beyond prototyping.

By applying the automatic programming paradigm, it is possible to eliminate most software maintenance at the implementation level. By providing computer support at all design phases, the transitions between these phases can be computer mediated. In this way it is possible to reduce communication problems and keep all participants in the same loop. By separating hardware dependent and hardware independent mechanisms and using a hardware dependent rule-base for optimization, efficient rehosting is possible, and the software lifecycle can be extended beyond the hardware lifecycle. Furthermore, this allows a smooth transition from simulator evaluation to in-flight testing.

The fact that all components are already being integrated with existing equipment allows us to already use systems developed for the SRS for research purposes. A big advantage of this approach is that in this way much experience can be gained with this new technology.

References

1. Angus, I.A. 'Software Integration: The Problems and Risks of Reuse' *Proceedings of the 9th AIAA Computing in Aerospace Conference* (1993), pp. 631-639.

2. Balzer, R., Cheatham, T.E., Green, C. 'Software Technology in the 1990's: Using a New Paradigm' *IEEE Computer* (November 1983) pp. 39-45.

3. Beadon, A.A. 'Requirements for Rapid Prototyping of Crew Station Displays' *SAE Aerotech* (1988) pp. 151-155.

4. Bentley, J. 'Programming Pearls: Literate Programming' *Communications of the ACM* (May 1986), Vol. 29, No.5, pp. 364-369.

5. Boehm, B.W. 'Software Engineering' *IEEE Transactions on Computers* (December 1976), Vol. C-25, No. 12, pp. 1226-1241.

6. Boehm, B.W. ' A Spiral Model of Software Development and Enhancement' *IEEE Computer* (May 1988) pp. 61-72.

7. Bordano, A., Lacovara, J.U., DeVall, R., Partin, C., Sugano, J., Doane, K., Compton, J. 'Cooperative GN&C Development in a Rapid Prototyping Environment' *Proceedings of the 9th AIAA Computing in Aerospace Conference* (1993), pp. 883-890.

8. Cleaveland, J.C. 'Building Application Generators' *IEEE Software* (July 1988) pp. 25-33.

9. Davis, A.M., Bersoff, E.H., Comer, E.R. 'A Strategy for Comparing Alternative Software Development Life Cycle Models' *IEEE Transactions on Software Engineering* (October 1988), Vol. 14, No. 10, pp. 1453-1461.

10. Fischer, G., Girgensohn, A., Nakakoji, K., and Redmiles, D. 'Supporting Software Designers with Integrated Domain-Oriented Design Environments' *IEEE Transactions on Software Engineering* (June 1992), Vol. 18, No. 6, pp. 511-522.

11. Green, C., Luckham, D., Balzer, R., Cheatham, T., Rich, C. 'Report on a knowledge based software assistant'. *Readings in Artificial Intelligence and Software Engineering* (1986) ISBN 0-934613-12-5 Morgan Kaufmann Publishers, Inc. CA.

12. Harel, D. 'Biting the Silver Bullet' *IEEE Computer* (January 1992) pp. 8-20.

13. Klein, M.H., Lehoczky, J.P., Rajkumar, R. 'Rate-Monotonic Analysis for Real-Time Industrial Computing' *IEEE Computer*, January 1994, pp. 24-33.

14. Lamerigts, T.T.H. *BSI: Basic Research Simulator System Interconnection* (1992) Master's Thesis, Delft University of Technology, Faculty of Electrical Engineering, Department of Telecommunication and Traffic Control Systems.

15. Levy, L.S. 'A Metaprogramming Method and Its Economic Justification' *IEEE Transactions on Software Engineering* (February 1986), Vol. SE-12, No. 2, pp. 272-277.

16. Lowry, M.R. 'Software Engineering in the Twenty-First Century' *Automating Software Design* (1991), pp. 627-654, AAAI Press. ISBN 0-262-62080-4

17. McCartney, R. 'Knowledge-Based Software Engineering: Where We Are and Where We Are Going' *Automating Software Design* (1991), pp. xvii-xxxi, AAAI Press. ISBN 0-262-62080-4

18. Plant, R.T. 'Software Development Process Model for Aerospace Systems' *Proceedings of the 9th AIAA Computing in Aerospace Conference* (1993), pp. 883-890.

19. Rich, C. and Waters, R.C. 'Automatic Programming: Myths and Prospects' *IEEE Computer* (1988), pp. 40-51.

20. Theunissen, E. 'The DELPHINS Tunnel-in-the-Sky program' *Proceedings of the Looking-Ahead Symposium* (1993), Amsterdam, The Netherlands.

21. Theunissen, E. 'The Development of the Delphins Display Design System' *Proceedings of the International Training and Equipment Conference* (1994), pp. 583-588. The Hague, The Netherlands.

22. Tindell, K.W., Burns, A., Wellings, A.J. 'Guaranteeing Hard Real Time End-to-End Communication Deadlines'. *Technical report* (1993), Dept. of Computer Science, University of York.

23. Turkovich, J.J. 'Automated Code Generation for Application Engineers' *Proceedings of the 9th Digital Avionics System Conference* (1990), pp. 616-627.

APPLICATION OF GRAPHICAL SOFTWARE GENERATION TECHNOLOGY IN SIMULATION OF ALL-DIGITAL TRANSPORT AIRCRAFT

AIAA-94-3412-CP

Shane M. Fookes and Nancy T. Heen
Boeing Commercial Airplane
PO Box 3707 M/S 19–MH
Seattle, WA 98124–2207
(206)–662–4959; FAX (206)–662–4404

Abstract

Over the years, airplanes have become increasingly dependent on digital systems. Largely because of this, the value of airplane system simulations has increased substantially and the simulation user community has broadened dramatically. To meet the growing demand on its services, the simulation development community must continue to pursue, and even create, new methods and technologies. Faced with this challenge, The Boeing Company's Flight Systems Laboratory (FSL) developed a Graphical Software Development (GSD) system with automatic code generation capabilities as a key part in its support of airplane simulation development, particularly development of the 777 airplane simulation. This paper provides an overview of Boeing's implementation of GSD technology; shows how it has been used to meet the demanding requirements of designing, simulating and testing the 777 airplane; and discusses the challenges that FSL has had to face because of the use of GSD.

Introduction

For Boeing, the scope of simulation has changed significantly. The simulation needs of airplane programs increased as the airplanes increasingly relied on digital systems. In 1980, there were few simulations. Those which did exist encompassed only "basic" airplane characteristics of the 727, 737, and 747 airplanes. By 1981, with the development of the 757, 767, and the 737–300, simulations of the Autopilot and Flight Management Computers (the first digital airplane systems) were added to the basic airplane simulations. The size of the simulation doubled with these additions. These simulations were used to support newly developed avionics "test benches". Thus, the user community of the simulation expanded. In 1984, the development of the 747–400 brought further changes to simulations at Boeing, and to FSL. Complex engines, displays, flap/slats, and warning/caution systems, among others, were added to the simulation. The number of systems and subsystems simulated increased substantially and the overall size of the simulation again doubled. The simulation now supported multiple test benches and an engineering simulator with a fully functioning flight deck.

The 777 airplane has been designed as Boeing's first all–digital aircraft. This has had an impact on FSL far greater than any previous change. Simulation size has jumped 5–7 times from the 747–400. In previous test benches, 1000–2000 signals made up the software/hardware interface, now every wire or signal in the airplane (30 to 50 thousand signals) is potentially available in the simulation. In addition to supporting stand–alone test benches and fully functioning flight deck cabs, the simulation also supports integrated test–benches, a Flight Controls Test Rig (or "Iron Bird"), and a Systems Integration Lab that contains all the avionics on the airplane.

To meet the 777 airplane development needs and schedule, FSL relied heavily on a number of new technologies, one of which was GSD. Although GSD was developed at Boeing before the start of 777 development, its use has been primarily in support of this airplane program.

Overview of GSD

As its name implies, the GSD technology provides the ability to create source code files graphically. It is used to greatly reduce errors, both interpretational and syntactical, and coding effort through a significant level of automation. A GSD diagram editor permits a user to create system diagrams via a graphical user interface employing familiar diagramming symbols. GSD utilities are then available to perform syntactic checking of individual diagrams, to output diagrams in documentation–ready formats, and to automatically create compile–ready source code. FSL developed a GSD system in–house to support simulation development activities. FSL's implementation is known as GSDS (Graphical Simulation Development System).

The diagram editor allows the user to create three types of diagrams: (1) flow control, (2) block diagram, and (3) dataflow. These diagrams completely describe the execution sequence, computational aspects, and data flow of a system. Each of the diagram types are described below.

Flowcharts are used by many software engineers to represent control flow visually. The GSDS diagram editor allows the engineer to not only create these diagrams graphically, but also to use them to organize the block diagrams and ultimately the simulation code. An example flow control diagram is shown in Figure 1. If a flow control diagram gets too complex, it can be decomposed into hierarchical components. The flow control diagrams, then, specify the execution of other flow control diagrams and of block diagrams.

The block diagrams form the bottom, or primitive, layer of diagrams. These diagrams depict the computations and simple functions needed to implement the system being modelled. They consist of elements such as single– and multi–variable inputs and outputs, summing junctions, gains, limits, quotients, and many other computational symbols. Block diagrams can also use table–lookup functions and user–defined icons that can represent any programmable function. An example block diagram is provided in Figure 2.

The dataflow diagrams show the flow of signals through the system. With these, the user can verify a signal's origin and path through the system. Signals can either be shown singularly or can be grouped together into vectors. An example dataflow diagram is shown in Figure 3.

The diagram editor provides a palette of standard diagramming symbols (Figure 4). The user can quickly create diagrams with a series of mouse clicks and movements. A location is selected on a diagram and then a symbol is chosen from the palette. The symbols are then interconnected simply by selecting the source and destination symbols. After a diagram is completed, a syntactical check is available to verify the diagram's correctness and to detect invalid feedback loops. In addition, the input and output data types for each symbol are verified for consistency and correctness for that symbol and missing inputs and outputs for each symbol are detected.

The signal (variable) data types are defined by one or more data dictionaries. Each entry in the data dictionaries consists of the variable name followed by its type, common block, and description, and value if it is a constant. The dictionaries may be edited while running the diagram editor or independently of the diagram editor. These dictionaries support a multi–user environment. Interface symbols between modelled systems are defined in the source system's data dictionary. The designer of the destination system then accesses the variables from that dictionary.

GSDS, using an database, handles each symbol internally as a construct whose primary attributes are its type and its input and output lists. The type is used by the code generator to determine which of a variety of coding templates to use in generating source code. It also defines what requirements are placed upon inputs and outputs to each symbol. Connections between symbols are handled as constructs whose primary attributes are the element from which the line originates (its source) and that at which it terminates (its destination). The code generator uses these connections to verify type compatibility and determine coding order.

GSDS usage in 777 program

At the onset of the development effort, the 777 program selected a number of new technologies to achieve the program initiatives. One of these was the use of GSDS for Specification Control Document (SCD) and simulation development. The fundamental reasons for this choice were: (1) simplified documentation: one document would exist for both the SCD and the simulation, (2) reduced cycle time for simulation development, (3) reduced number of people required for SCD and simulation development, (4) reduced variations introduced into simulation code that result from the different programming styles of code developers, and (5) achieved a level of Line–Replaceable Unit (LRU) simulation fidelity never before achieved in a Boeing airplane, in a shorter period of time. GSDS met the program expectations in each of these areas. However GSDS usage presented FSL with unforeseen challenges during the 777 simulation development period. The following two sections describe the benefits realized by the 777 program and the challenges that FSL addressed as a result of the GSDS usage.

Benefits to 777 Program

SCD and Simulation Development

The use of GSDS altered the SCD and simulation development practices for a number of LRU and simulation developers. In past airplane programs, without GSDS, SCDs for the LRUs and airplane systems were designed by engineers and then developed either on paper or on a stand–alone, computer–based drafting system by technical support personnel. The SCDs were, and remain, the primary documentation for vendor support. In addition, SCDs were used to develop simulation documents from which simulation code was hand–developed by FSL engineers.

With the increase in both the number of digital systems, and in their complexity, a more efficient method was needed for SCD and simulation development. GSDS provided this efficiency. With GSDS, the SCDs were created using the GSDS diagram editor. The simulation code was then generated directly from those diagrams with the code generator. Because the code was generated directly from the SCD, a number of benefits were realized. These benefits are described in the following paragraphs.

Because SCD development and simulation development occurred simultaneously, system simulations actually pre–dated SCD releases. Thus, the simulation could be used to validate early designs and the designs were modified accordingly. The initial SCD releases were improved as a result.

GSDS offered an ease of use which impacted the system simulation designs. The effort required to code a model was now a fraction of what it was without GSDS; consequently model detail and precision increased dramatically. Certain LRU and airplane system details were emulated in the software models. The overall airplane simulation became a collection of detailed truth models rather than broad mathematical models. In addition, the simulation model interfaces were created from data dumps of the actual airplane Interface Control Drawing (ICD) database. Thus, the resulting simulation closely mapped the actual airplane digital design. Because of this, the LRU (or airplane system) design has been tested at a more detailed level. Since the simulation directly mapped the SCD, system design validation was more exact. In addition, the software models became interchangeable with their hardware equivalents (commonly known as 'plug–compatible'). Thus, in an integrated hardware test environment, the time lost to non–functioning LRUs has been reduced by swapping in the software models.

FSL was now involved earlier in the airplane design cycle. The system designers needed simulation turn–around times from FSL reduced. The GSDS code generator, along with other FSL developed tools, provided the needed turn–around times. Response times for FSL were reduced up to 75%. Figure 5 shows example FSL response times with and without GSDS.

An additional benefit derived from the use of GSDS was the simulation code documentation. The code was now fully described by the GSDS block diagrams. This eased the time required to learn a system when new people were added to the project or when someone from a different area required system knowledge. Also, system designers now knew exactly what was coded in the simulation.

Simulation Common Models

One of the most important benefits that the 777 program realized because of GSDS was the ability to economically create a set of simulation software models that met a common set of requirements. These models became known as Simulation Common Models (SCMs) and became instrumental in developing software for the 777. Though not all the SCMs were produced with GSDS (about 60% were), the benefit of GSDS usage was significant on the overall development of the SCMs.

SCMs are simulation models of LRUs and airplane systems that were developed and released concurrently. The same models were then used in multiple environments. They were used by (1) 777 system developers to refine and validate their designs; (2) 777 designers to generate data documents for shipment to Training Simulator vendors; (3) flight deck design engineers in Flight Deck Cabs for cockpit design and airplane marketing; and (4) avionics testers in test benches, and other hardware test environments for LRU integration and validation. Because all the environments used a common set of software models, a higher level of individual airplane system design validation and airplane system integration occurred.

Since the models were used in a number of environments, the requirements against the models increased dramatically. The level of complexity required for the models also increased. In addition, the demands on the model developers increased so that the multiple test environments remained satisfied. GSDS provided the technology to achieve the higher level of complexity and to satisfy the demands from the test environments within a period of time and with a level of manpower that was economically acceptable for the 777 program.

Challenges to FSL

As with most new technology, the implementation of GSD has required changes to existing simulation development processes and to existing resources. FSL was faced with major changes in software configuration control, simulation development methods, disk space, and real–time computers. The effect on each of these areas is described in the following sections.

Configuration Control

The key elements in configuration control are: (1) determine what is to be controlled, known as the configuration item, and (2) determine how the configuration item will be controlled. With software, the configuration item is typically source code routines since they are the root elements of a software model. For a software model built with GSD, source code is no longer the configuration item. Instead, the GSD diagrams are considered the configuration item since they are now the root element of the software model.

At the inception of the 777 simulation development, FSL decided to make each individual diagram a configuration item. Since the diagrams are stored in ASCII format, a source control system could be utilized. FSL implemented the UNIX Software Code Control System (SCCS). As the simulation development progressed, the FSL engineers found using SCCS for diagram controlling was cumbersome and required ever escalating disk space. Two reasons were cited for these problems. First, SCCS was not readily accessible from the GSDS diagram editor. Thus, to maintain tracking of all changes, an engineer had to check out a diagram, enter the diagram editor, make the changes, exit the editor, and check the diagram back in. For a high volume of changes, this is burdensome for the engineers. Second, source control systems like SCCS store the differences between updated files and their originals rather than the entire updated files; this conserves disk space. This feature was nullified when SCCS was used with GSDS diagram files since even the smallest change to a diagram could result in a large change to the ASCII file.

Even though SCCS provided the tracking necessary to control the GSDS developed softwared, it was inefficient to implement and use. Because of this, FSL instead made the models themselves the configuration items rather than the GSDS diagrams.

The concept of each software model being a configuration item was accomplished with two key elements: the UNIX directory structure and FSL produced tools. The UNIX directory structure essentially provided a place holder for the different types of items that made up a total model. Each model had used a common directory "tree" (Figure 6). Within each model's directory tree were different sub–directories, one of which contained strictly GSDS related files, another source code files, and another object files. Each model had a directory tree in a development area and an identical tree in the configuration controlled release area. FSL then created tools to update and release models in the configuration control area at scheduled intervals.

Although inefficiency still exists, FSL is striving to continuously improve this configuration control scheme to provide a consistent, efficient configuration management for GSDS generated software models.

Simulation Development Practices

The use of GSDS changed simulation design methods for The Boeing Company. Prior to GSDS, simulations were hand–coded from SCD specifications provided by airplane system design engineers. The simulation engineers interpreted the specifications and developed the simulation code. A large portion of simulation development time was then devoted to resolving syntax and coding errors found while attempting to compile and link the code into a simulation. The simulation engineers would then produce data to verify that what was coded matched the original specifications. Once this was verified, the system design engineers could use the simulation to validate their design. This process is shown in Figure 5. This figure also shows how the simulation development process was altered considerably with GSDS. The design engineer and simulation engineer now developed the SCD with the GSDS diagram editor. The design engineer either directly entered design changes or provided red–lined GSDS diagrams to the simulation engineer. The simulation code was then generated from the diagrams with the GSDS code generator. The time spent debugging syntax errors was eliminated. Time spend in simulation design verification was also eliminated. Overall, it's estimated that a 25% revision to a simulation model with 50,000 lines of code takes 75% less time to implement when GSDS is used than when it is not.

Although this process improvement brought benefits to the design engineers, it brought new challenges for FSL. Because the simulation code was generated from SCD diagrams, the design engineers became the de facto simulation code designers and developers. The problem that surfaced was the SCD developers were more concerned with their diagrams conforming to SCD diagram etiquette than with the quality of the simulation code generated. The diagrams were often designed in ways that produced inefficient code. FSL's role in code design was reduced to making diagramming recommendations and improving the code generator wherever possible.

Another challenge for FSL was minimizing the impact of changes to the GSDS tool itself on the SCD and simulation development. With the significant GSDS usage in the 777 SCD and simulation development, the GSDS designers became an important contributor in the design loop. The GSDS design was still forming at the beginning of the 777. In addition, the size and scope of the 777 development at times overextended the GSDS design. As a result, the GSDS problem resolution cycle time impacted design milestones.

Disk Space

Another factor affected by the implementation of GSDS was disk space. Disk space was considered by many to be a non–issue since it was typically a 'cheap' resource. The increase in the amount of disk space required by models developed with GSDS was far beyond projections.

Individually, GSDS diagrams are not large consumers of disk space. A GSDS diagram could range anywhere from 1 Kilobyte (Kb) to 120 Kb of disk space, with an average diagram taking about 20 Kb. However, a common 777 LRU simulation model contains approximately 300 diagrams. Thus the size of the package of diagrams is 6 Megabytes (Mb). The GSDS database, constructed from the diagrams and dictionaries uses an additional 4.5 Mb. The code generator creates approximately 23,000 software lines of code (sloc) from the 300 diagrams. This code uses 1.7 Mb. When this code is compiled, the binaries consume an additional 0.8 Mb. The total disk space consumption for the example GSDS–developed simulation model is 13 Mb. A hand–developed simulation model with a similar number of sloc consumes about one–fifth the disk space.

The numbers described above made up a fraction of the total amount of disk space necessary for development and release of the set of SCMs. Every airplane system simulated had a unique disk space requirement based on the needs of the customers of that model. The typical disk space requirement was 3–4 times that of previous airplane simulation development. Each SCM release required 300 Mb. The multiple SCM customers used releases at different rates. As many as 6 SCM releases were concurrently available.

None of FSL's computer systems supported physical disks large enough to meet the disk space usage mentioned above. As a result, network of software links was established between available disks to create illusion of contiguous disk space of the size required by the SCMs. Several problems surfaced with this implementation: (1) machine performance was degraded, (2) the number of system links required soon exceeded the maximum allowed on the computer systems, and (3) the number of disks required to meet the total disk requirement exceeded the number of physical mounts available on the computer systems.

Other steps were taken to help manage the unexpected disk space needs. Development groups were given responsibility for managing their own disk usage with a budgeted amount of disk space. Thus cleanup of unnecessary disk usage was peer induced. Also, configuration control was centralized to one group which centralized the task of managing the total set of SCM released software. Eventually, though, FSL was forced to expand its disk resources to meet the unexpected demand.

Real–Time Performance

The requirements were demanding for the 777 airplane simulation and its individual software model components. The 777 program desired a level of comprehensive LRU testing and integration never before achieved by the company and required it earlier in the airplane development cycle. Simulation requirements included stringent frame times (10, 5 and even 1 milliseconds), latency limits, airplane ICD incorporation, software/hardware interchangeability, and simulation update rates which support hardware LRUs.

To support the demanding real–time requirements, FSL chose the state–of–the–art Harris NightHawk 4800 multiprocess real–time system in 1990. This system provided up to 10 times the compute power of the real–time systems in FSL at the time. Table 1 shows the hardware specifications of the NightHawk 4800.

TABLE 1 Harris NightHawk 4800 Specifications

CPU

Longword Length	32 bits
Processors	one or two per board (maximum 8 per system)
System Bus	Harris VME
Physical Address Space	4 GB
Floating Point Accelerator	On–chip
Memory Management Unit	Motorola MC88200 CMMU, 25MHz 4 KB memory pages 56–engry Page Address Translation Cache
Cache Memory	Motorola MC88200 CMMU, 25 MHz; both write–through and copy–back algorithms used; concurrent address translation and cache access.

MEMORY SUBSYSTEMS

Maximum Physical Memory	192 MB
Global Memory	8 to 128 MB
On–board Memory	Available in increments of 8–, 16–, 32–, and 64–MB per board; 0, 1 or 2 boards per system

INPUT/OUTPUT

I/O Bus	8–,19–, or 21–slot Harris VME, superset of VME with parity checking and synchronous burst mode performance enhancement; triple–height Eurocard.
I/O Bus Performance	40 MB per second (with memory expansion); 10 MB per second (without memory expansion).

COMMUNICATIONS

Local–Area Network	Ethernet (IEEE 802.3), TCP/IP
Wide–Area Network	X.25 TCP/IP packet switching; DDN conformance

GSDS, as mentioned earlier, was instrumental in FSL meeting functionality and fidelity simulation requirements. However, as the simulation developed, its size grew far beyond expectations. The fact that the diagrams were created more for SCD etiquette than for real–time performance had a large impact. Also, system designers tended to overdesign the simulation components with GSDS, because of the ease of use it provided. As a result, frame time requirements were relaxed and more CPUs on the Night Hawk were used to run the simulation – thereby increasing latency in the simulation. Additionally, the simulation overextended the dynamic memory on the machine and the number of symbols in the simulation overwhelmed the machine I/O capability. FSL was faced with either improving the hardware/ software configuration or scaling back the simulation.

Thus far, a combination of hardware and software changes have been implemented to improve the real–time performance of the simulation. With hardware, a series of CPU upgrades, memory additions, operating system updates, and compiler improvements were installed until the next generation multiprocess machine was available – the NightHawk 5800 series. Beginning in December 1993, the major test systems in FSL began converting to the 5800. Because the simulation was generated directly from SCDs, software improvements were limited to multirating the simulation models, improving the GSDS generated code, and optimizing the compiled code.

With these improvements, FSL was able to meet the program requirements, though some of the original requirements were relaxed. The simulation did not exceed latency limits, met the relaxed frame time requirements, and supported hardware LRU update rates. FSL provided simulation models which, when integrated with the airplane ICD, were compatible with their hardware counterparts.

Summary

The simulation community has seen, and will continue to see, increasing demands on its products, especially with the proliferation of digital systems on aircraft. One way we in FSL have sought to meet this challenge is through the implementation of GSDS. The fact that FSL has been able to meet the stringent demands of the airplane program for simulation capability and availability is largely because of the widespread use of GSDS for SCD and simulation development. At the same time, this technology has challenged FSL in the areas of software configuration control, simulation development, disk space resources, and real–time performance.

The major complication in simulation development introduced by the use of GSDS was the trade–off between SCD diagram etiquette and simulation performance. Most often, the LRU design engineers deemed diagram etiquette more important than efficient simulation code. As a result, the performance of the LRU simulations was impacted significantly.

The development of the 777 airplane has shown that implementation of the graphical simulation development technology is a critical step toward total integrated development of an all–digital airplane.

References

Graphical Simulation Design System User Manual, D6–54697, Nov. 1988, K.C. Babb et al, The Boeing Company.

Night Hawk Series 4000 Architecture Manual, 0830040, Oct. 1990, The Harris Corporation – Computer Systems Division.

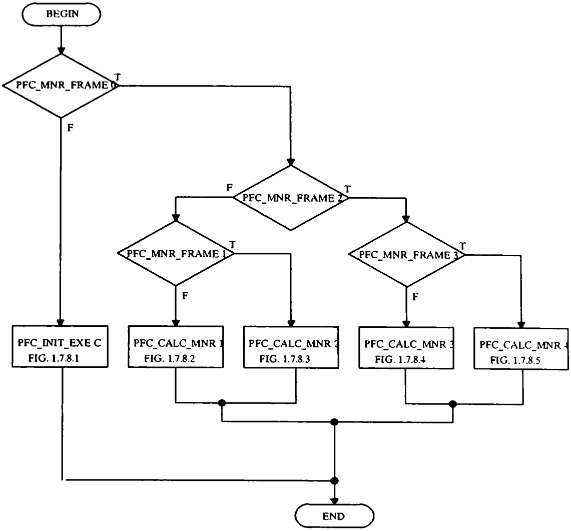


FIGURE 1: Example GSDS Flow Control Diagram

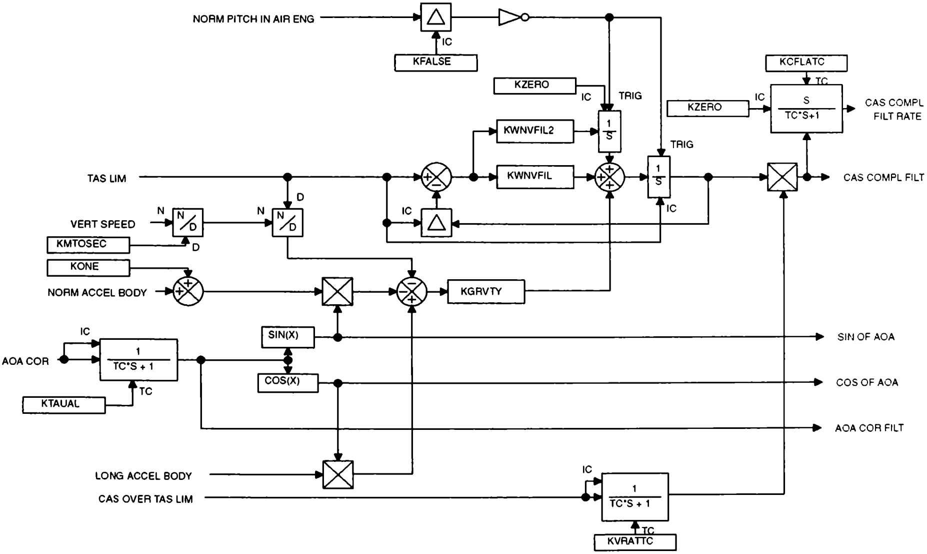


FIGURE 2: Example GSDS Block Diagram

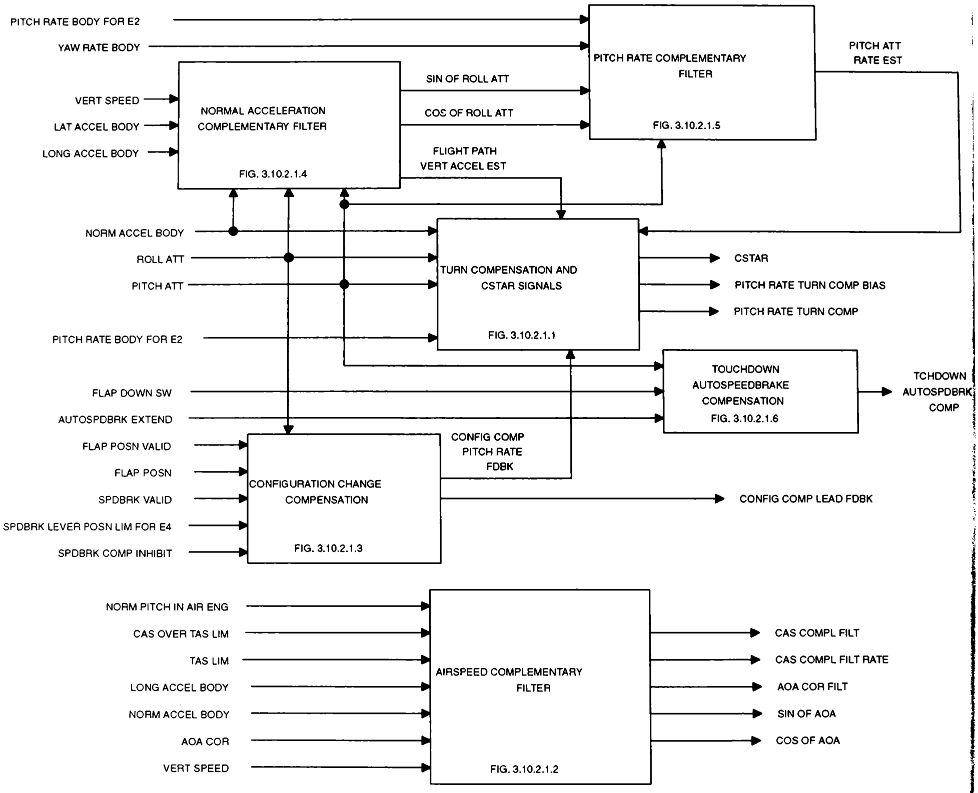


FIGURE 3: Example GSDS Dataflow Diagram

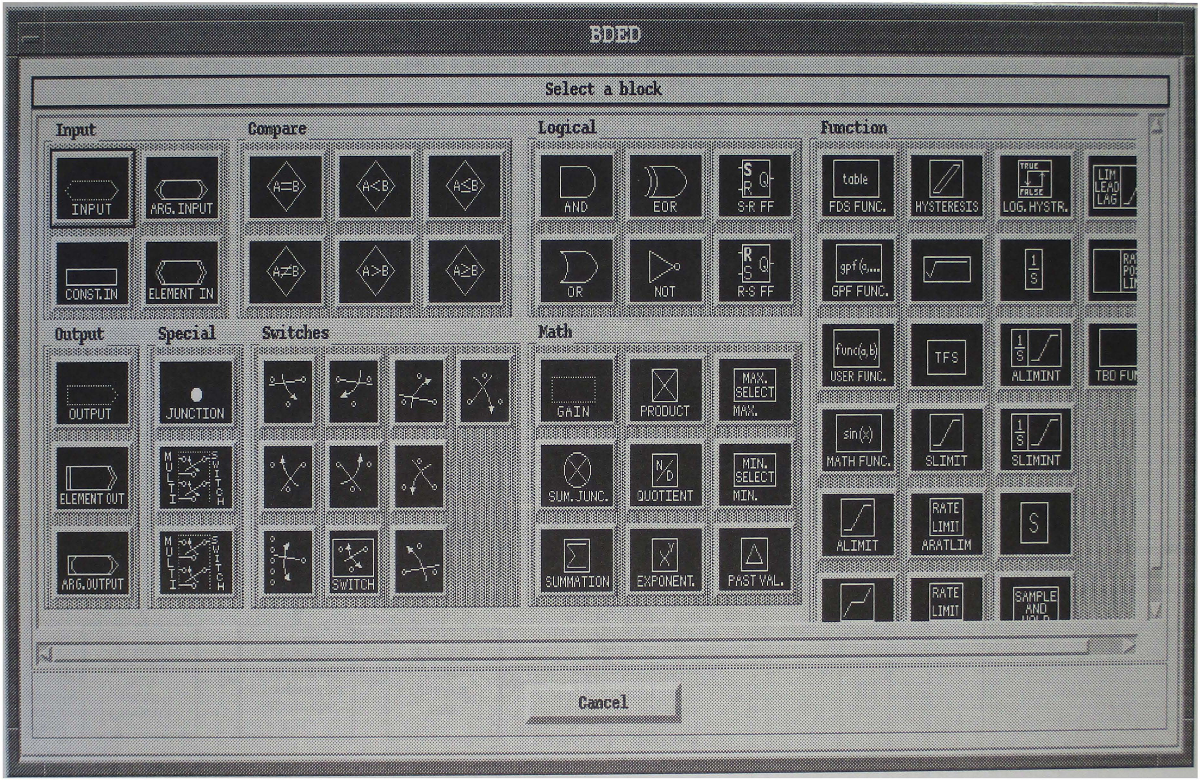
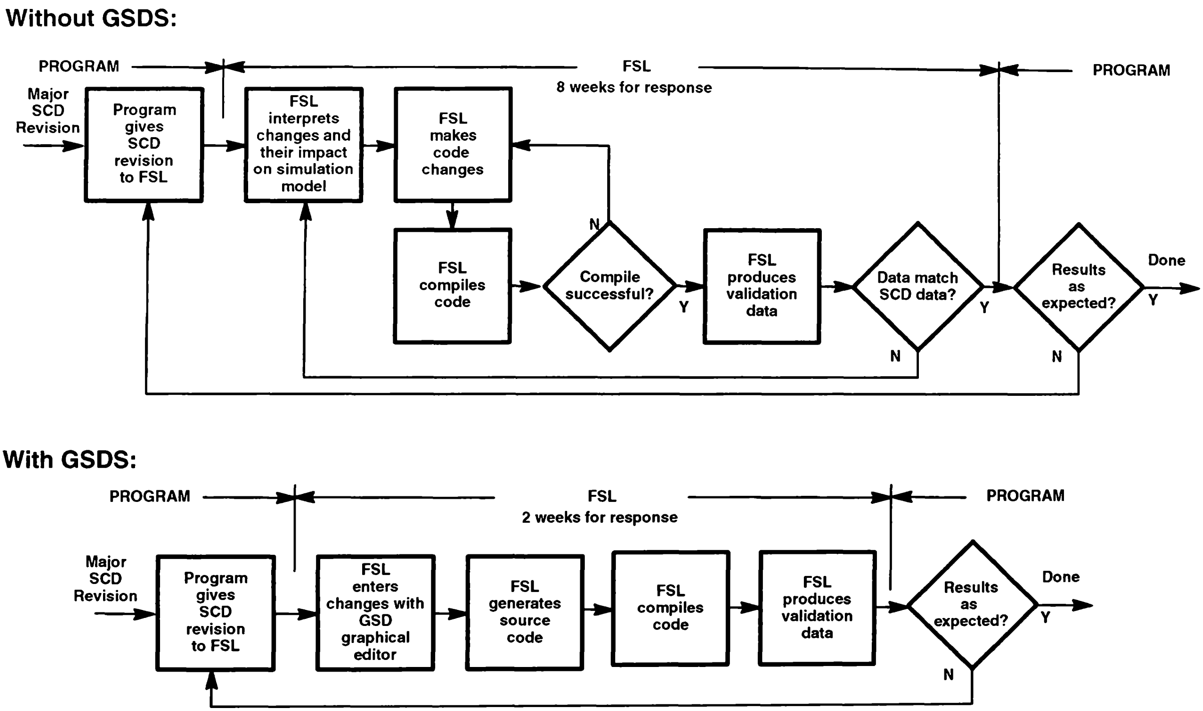


FIGURE 4: GSDS Block Diagramming Symbol Palette

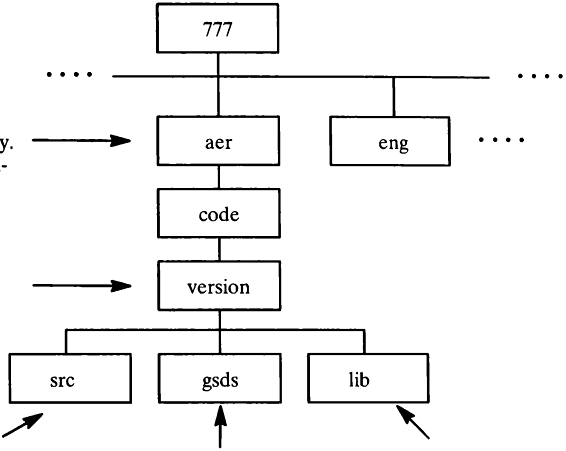


Without GSDS:

With GSDS:

PROGRAM
FSL
2 weeks for response
PROGRAM
Major SCD Revision
Program gives SCD revision to FSL
FSL enters changes with GSD graphical editor
FSL generates source code
FSL compiles code
FSL produces validation data
Results as expected?
Y
N
Done

Figure 5: Simulation Model Development Process



Each software model has a directory. The sub–directory structure is common for all models.

Each Subset contains multiple versions, each configuration controlled

The 'src' directory contains all the source files for the model

The 'gsds' directory contains all the GSDS specific files.

The 'lib' directory contains all the compiled object files for the model.

FIGURE 6: Unix Directory Structure for GSDS Model Configuration Control

AIAA-94-3413-CP

SIMULATION OF A REDUNDANT DIGITAL ASYNCHRONOUS FLIGHT CONTROL SYSTEM IN A MULTI–PROCESS ENVIRONMENT

Benton L. Parris and Les R. Fader
Boeing Commercial Airplane
P.O. Box 3707 MS 19–MJ
Seattle WA 98124

Abstract

A digital flight control system with redundant Line Replaceable Units (LRUs) and data busses which all run asynchronously with respect to each other could not be accurately modeled with traditional methods. Simulation of such a system requires variable time steps and a much higher computational load than single–thread simulations. This paper describes a variable time step method and special task schedulers which manage the execution of asynchronous tasks. Also described is the methodology of distributing the tasks in a parallel processor environment. The ability to alter the parameters which effect the asynchronous nature of the system allows one to study aspects of digital flight control systems that could not be done in traditional simulation environments or in hardware bench studies.

Introduction

Traditional airplane simulations consist of a flight control system, control surface and actuator dynamics, other subsystem models, aerodynamics models, airplane dynamics, and a model of the flight environment. Most of the simulated systems and the flight environment are continuous (or analog) and are modeled using ordinary differential equations. The flight control system is typically digital with multiple sample rates. These types of aircraft simulations are used extensively to study and refine aircraft and flight control systems designs.

Digital flight control systems usually contain redundant signal paths through different LRUs that carry pilot commands to control actuators. These redundant signal paths are typically not modeled because their contribution to the overall airplane performance in a healthy system is minimal. Including them often prevents a simulation from running in “real time” due to increased computational loads.

A major design consideration for digital flight control systems is the mechanism used to control data flow and the scheduling of events. Systems can be designed with a centralized controller that schedules all events in the system. Distributed systems may have many self contained controllers that only schedule events in a limited part of the system. In distributed systems with multiple controllers that are strictly autonomous, identical events can happen at different times in different signal paths. The ability to control these differences in event times is critical so that the simulation user can modify and observe the asynchronous behavior of the system.

This paper describes software and methods that allow careful control of event timing in the simulation of an airplane with a redundant asynchronous flight control system.

Scope and Methods of Approach

Three major topics are covered in this paper: (1) a set of variable time step schedulers used to control asynchronous events, (2) user specified data that defines the asynchronous behavior of the system, and (3) the distribution of models across multiple simulation computer processors to improve simulation performance.

Flight Control System Modeled

Figure 1 illustrates the flight control system being modeled. This system utilizes redundant paths of digital signals. The devices which process these signals are called Line Replaceable Units (LRUs). The signals originating from the pilot control inputs and various aircraft sensors are transmitted on multiple asynchronous data busses and read from the busses into multiple flight control computers. The outputs of the flight control computers are transmitted on the asynchronous data busses and read from the busses into actuator controllers which close the analog loop around the control surface actuators. Each LRU has its own clock that determines when events happen within the LRU. Each data bus transmitter (one for each LRU) has its own clock that determines when it is permissible to transmit. The system is asynchronous because no central controller exists to synchronize LRUs or transmitters.

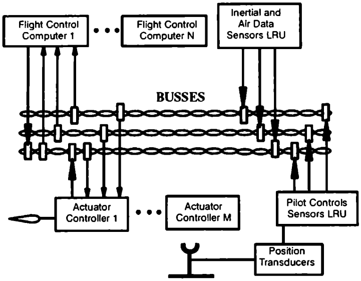

Flight Control Computer 1
• • •
Flight Control Computer N
Inertial and Air Data Sensors LRU
BUSSES
Actuator Controller 1
• • •
Actuator Controller M
Pilot Controls Sensors LRU
Position Transducers

Figure 1 – Flight Control System Modeled

Modeling Events

The two basic types of events modeled are those within an LRU (referred to as tasks) and those that happen between LRUs (referred to as transmissions). All tasks within a single LRU are controlled by that LRU's clock and are executed in a defined sequence. No central controller exists to synchronize LRU or bus transmitter clocks. Therefore, tasks within one LRU are asynchronous with respect to tasks in another LRU and transmissions on one bus are asynchronous with respect to transmissions on another bus.

In an asynchronous system, events do not occur in a predetermined sequence. A mechanism must be built into the simulation to allow for all possible sequences of events. This mechanism must provide repeatability or changeability of the sequence of events based upon predefined parameters. A user must have control over this set of parameters to be able to study effects of variations in asynchronous event timing.

The two basic sets of parameters controlled by the user in this simulation are clock rates and start times associated with each LRU and transmitter. Controlling the values for these parameters gives the user complete control over the sequence of events.

A common time must exist to compare results of asynchronous events. All events in typical simulations are directly tied to a common time variable. In this simulation, an independent reference time is defined that is external to all LRU and transmitter clocks and is referred to as **base time**. All LRU and transmitter clock rates are defined relative to base time.

Figure 2 illustrates LRU task and transmitter event timing, the parameters controlled by the user and their interrelationship.

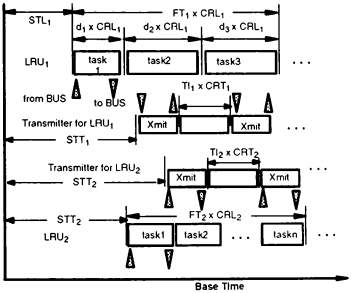


Figure 2 – LRU Task And Transmitter Time Lines

Each LRU clock and transmitter clock has a start time (STL_m and STT_m) and a clock rate (CRL_m and CRT_m) set by the user. A start time indicates when an LRU or transmitter will begin functioning relative to base time. The clock rate is a ratio of clock time to base time. For example, an LRU clock rate of 1.0 results in time on that LRU being identical to base time and a clock multiplier of 1.01 would force that LRU to run 1% faster than base time. All LRUs and transmitters may be set to start at different base times and run with different clock rates.

The user also specifies the number, order and duration of tasks within each LRU. The task duration (d_i) is the time allowed to execute that task as measured by the LRU clock. The sum of durations for all the tasks in that LRU represent the frame time for that LRU (FT_m). The continuous aspects of the simulation, like the airplane and control actuator dynamics, are considered LRUs with a clock rate of one.

Similarly, for each transmitter the user must specify what messages are to be transmitted, the order in which messages are to be transmitted, the wordstrings that make up each message, the duration of or allowable time to transmit each word string, the time between transmissions (TI_m), and any other transmitter protocol parameters.

Control of Time and Events

Control of events in simulations is based on the control of some time base and when events occur in that time base. Time for simulations of analog systems is controlled by a fixed time step. Procedures for integrating ordinary differential equations with a fixed time step are well known. The selection of a fixed time step is a trade between simulation execution speed and the fidelity of the dynamics being modeled. For simulations of digital systems, it is typical to select a fixed time step as a function of sampling frequencies. In both types of simulations it is possible to select a fixed time step that will model the necessary dynamics and discrete events while maintaining an acceptable simulation execution speed.

In this simulation, the interval between asynchronous discrete events varies as time advances and may be as small as one nano second. A fixed time step would need to be small enough to see all events. A one nano second fixed time step would result in a simulation execution speed too slow to be practical. Therefore, a variable time step with a resolution of one nano second was chosen. For example, the time step may at times be one nano second and other times be 10 milliseconds depeneding on the interval between asynchronous events. It should be noted that although base time changes with a variable time step, tasks within an LRU use a fixed time step for integration. This fixed time step is the frame time of that LRU relative to the LRU clock.

This simulation incorporates two types of schedulers to control events. A task scheduler controls the advance of base time and when LRU tasks run. A bus scheduler controls when the bus model runs relative to the task scheduler. The bus model controls when each transmission occurs according to bus protocol and timing parameters.

Task Schedulers

This simulation is partitioned into separate processes. Each process contains a copy of the task scheduler that will execute tasks for one or more LRU.

The coordinated effort of all task schedulers is to execute each task in each LRU in an order that represents the start times, clock rates, and task durations that were specified by the user.

Coordination between all task schedulers is accomplished by posting values to areas in common (or shared) memory. The two key areas are the one reserved for the **posted table** and the one reserved for the **task table**.

The **posted table** contains the name of the task that is to run next, the time that task is to begin (**posted time**) and the scheduler associated with that task.

The **task table** is used to determine which task of all tasks in the system to post in the posted table. An example of the task table is shown below.

Task Table

LRU	Task	Time	Scheduler
LRU_1	$fltctr_3$	0.018	1
LRU_2	$fltctr_1$	0.024	0
LRU_3	$fltctr_5$	0.015	5
LRU_4	$actctr_3$	0.017	3
LRU_5	$actctr_1$	0.016	9
LRU_6	$sensor_2$	0.020	3
⋮	⋮	⋮	⋮
BAP	aero	0.040	0

The table has one entry for each LRU in the system. An entry contains the task to run next in that LRU, the time (in terms of base time) that task is to begin, and the scheduler which will execute that task.

The task scheduler in each process continually monitors the scheduler parameter in the posted table. When its unique value is posted it begins the execution phase of the posted task. The scheduler coordinates with the bus scheduler, updates base time and determines the task and time to run that should be posted next during the execution phase.

Execution Phase

The task schedulers coordinate communication between tasks and the bus model during the execution phase illustrated on Figure 3.

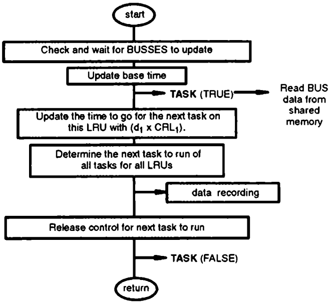

start
Check and wait for BUSSES to update
Update base time
TASK (TRUE)
Read BUS data from shared memory
Update the time to go for the next task on this LRU with (d_i x CRL_i).
Determine the next task to run of all tasks for all LRUs
data recording
Release control for next task to run
TASK (FALSE)
return

Figure 3 – Execution Phase

A task scheduler begins its execution phase by waiting until the busses have come up to the current posted time. Once it is determined that the busses are up to date, base time is updated. Then that part of the current task which performs communication with the busses is executed. The bus model is therefore not being executed while a task is reading bus data. This insures proper coordination of communications between the bus and tasks.

After the bus communication portion of the task is completed, the task scheduler determines the next task that will be run on the current LRU. The task scheduler then updates this LRU's row in the task table with this task and it's time to run. The time to run is determined by multiplying the duration of the task by the clock rate for the LRU (d_i x CRL_m).

Then the table is scanned to determine the next task, across all LRUs, that is to be executed. Data recording for this process is executed and the next task, time to run, and corresponding task scheduler are posted.

Posting the next task allows the associated task scheduler to begin the execution phase and allows the bus model to run.

The remaining computations in the current task are executed after posting.

This structure of the execution phase assures that the bus model and a task do not try to access identical shared memory at the same time. The structure also takes advantage of parallel processing environment discussed later.

Bus Scheduler

The bus model and bus scheduler run in the same process. The bus scheduler coordinates the execution of the bus model in relation to the task schedulers. Because transmissions and bus protocol sequencing happen at a much higher rate than typical LRU tasks, we let the bus model run freely between LRU tasks resulting in improved simulation speed. The bus scheduler coordinates with task schedulers using the dynamically changing value of posted time. The bus is allowed to execute all transmissions and all protocol sequencing from the current base time up to the posted time.

Bus Model

The bus model controls bus transmissions according to the transmission timing and parameters which define the sequencing protocol. Figure 4 illustrates the two major functions of the bus model. The first function, *protocol sequencing*, determines the next transmitter on each bus and when to begin transmitting. The second function, *message transmission*, transmits the next message on each bus. Transmissions will be interrupted if the time to transmit a message exceeds posted time.

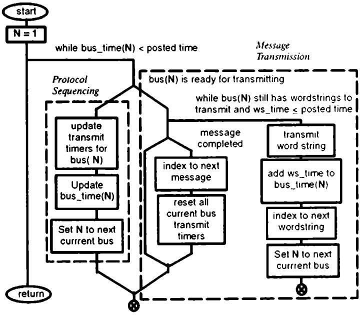

start
N = 1
while bus_time(N) < posted time
Message Transmission
Protocol Sequencing
bus(N) is ready for transmitting
while bus(N) still has wordstrings to transmit and ws_time ≤ posted time
update transmit timers for bus(N)
Update bus_time(N)
Set N to next currrent bus
message completed
index to next message
reset all current bus transmit timers
transmit word string
add ws_time to bus_time(N)
index to next wordstring
Set N to next currrent bus
return

Figure 4 – BUS model

The user defined bus protocol parameters determine the next active transmitter. The purpose of these parameters is to avoid two or more transmissions on the bus at the same time and to set transmission frequency of each transmitter. For example, an ARINC 629 bus has the synchronization gap, terminal gap, and transmit interval as bus protocol parameters. The synchronization gap and terminal gap force a specified "quiet" time on the bus between each transmission. The transmit interval determines the frequency of transmissions, that is, the time between transmissions of a single transmitter. The next transmitter to transmit is the first to satisfy all three of the protocol parameters.

Figure 5 illustrates the updating of the busses. One bus at a time is advanced through the required protocol sequencing up to where a message is to be transmitted. If the transmission start time is less than posted time then transmission of the message begins.

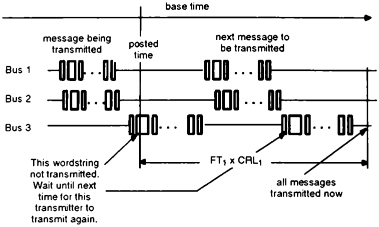

base time
message being transmitted
posted time
next message to be transmitted
Bus 1
Bus 2
Bus 3
This wordstring not transmitted. Wait until next time for this transmitter to transmit again.
FT_1 x CRL_1
all messages transmitted now

Figure 5 – Updating Busses

Each transmission is a message containing a group of wordstrings. A wordstring is an array of variables. The time required to transmit each wordstring across the bus is specified by the user. Only one message is transmitted on a bus at a time. When a message is finished a prescribed "quiet" time must elapse before another message may be transmitted.

While a message is being transmitted, time for that bus is increased by the wordstring transmission time as the word is transmitted. If the next word string to transmit will exceed posted time the next bus is selected. If the message completes before posted time the current bus is advanced up to the next transmission time.

It is important to note that each transmitter is autonomous to all other transmitters on that bus and all transmissions on a single bus are considered asynchronous. In a multiple bus system, all transmissions are asynchronous to all other transmissions.

Once all busses have been advanced to posted time, the bus model returns to the bus scheduler.

Simulation Mode Control

This simulation utilizes a standard definition of operational modes for the simulation. These modes are defined as follows:

IC – This mode is for setting all models to their initial conditions. It is also the mode in which trimming is performed. Base time is zero in this mode.

COMP – This is the mode in which the dynamic models with time dependent integrators are running and base time advances.

HOLD – This mode is for "freezing" the simulation in time. This stops execution of all models and associated updates of the data.

TRIM – This is not a separate mode but a state of the IC mode in which software is run to null airplane accelerations.

The model code and task schedulers execute in different ways depending on which mode is in effect.

Although the heart of the models are executed in COMP mode, what and how things are performed in IC mode is critical to generating a valid starting point for COMP mode.

Mode Transitions

Since the models can be executed in different processes on separate CPUs, a mechanism must exist to insure that each model does not begin its IC or COMP calculations until all models are ready to begin that mode. One model must not be in the middle of IC calculations while another is doing COMP calculations. The models must "synch–up" before beginning their calculations in either mode.

Figure 6 illustrates how the task and bus schedulers control execution of all models during the transition from IC to COMP and the transition from COMP to IC. Individual bits in the *icdone* and *icmode* words and individual elements of the *ictoco* and *cotoic* arrays are assigned to a unique scheduler. The *icdone* and *icmode* words and *ictoco* and *cotoic* arrays all reside in shared memory so that they are accessible by all processes.

Transition from IC to COMP

While in IC, each scheduler clears a bit in the *icdone* word and sets an element of an integer array called *ictoco* to one.

When the scheduler observes that COMP is true (COMP mode is in effect for this process), *icmode* not equal to zero (not all processes have reached this

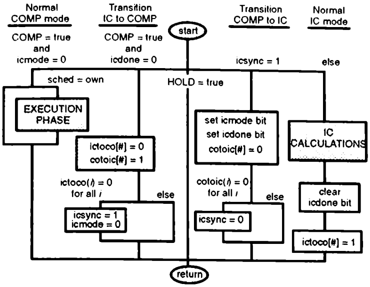


Figure 6 – Mode Transition Synchronization

point), and *icdone* equal to zero (all processes have completed the IC "synch–up") it sets its *icmode* and *icdone* bits and sets its element of the *cotoic* array to zero.

The scheduler then waits until all processes have set their element of the *ictoco* array to zero. This indicates that all processes have reached this point and so the *icmode* word is cleared and the *icsync* flag is set. This allows the processes to begin their COMP execution.

Transition from COMP to IC

When the scheduler observes that IC is true (IC mode is in effect for this process) and *icsync* is equal to zero (just out of COMP and processes not all here yet) it then waits until all processes have set their element of the *cotoic* array to zero. This indicates that all processes have reached this point and so the *icsync* flag is reset. This allows the processes to begin their IC execution.

Task Scheduler and Simulation Modes

Figure 7 illustrates the operation of a task scheduler in IC, HOLD, and COMP modes.

During IC mode each copy of the task scheduler in each process runs independently from the others and does not use the posted table in shared memory to determine when and what to execute. The task scheduler first zeros base time. To prepare for COMP mode, it initializes the task table and posts the next task, time to run and corresponding task scheduler. The task scheduler then executes each of its tasks, first for bus communication then for the remainder of task computations. The data recording function is then invoked.

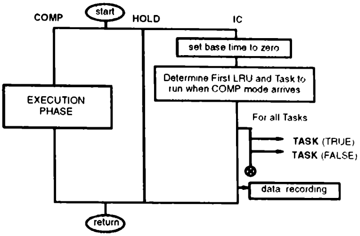


Figure 7 – Task Scheduler and Simulation Modes

The task scheduler does nothing in HOLD mode.

The COMP mode is the main mode of operation and the behavior of a task scheduler in this mode is described in the Control of Events section above.

Bus Model and Simulation Modes

Figure 8 illustrates the operation of the bus model in IC, HOLD, and COMP modes.

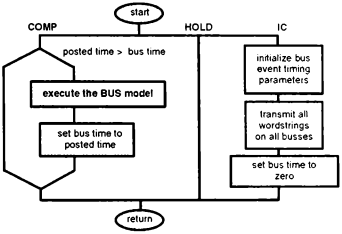

start
COMP
HOLD
IC
posted time > bus time
execute the BUS model
set bus time to posted time
initialize bus event timing parameters
transmit all wordstrings on all busses
set bus time to zero
return

Figure 8 – Bus Model and Simulation Modes

During IC, the bus model initializes the event timing and bus protocol parameters. It then transmits all wordstrings on all busses without coordinating with the task scheduler. It also sets the bus time to zero.

The bus model does nothing in HOLD mode.

In COMP, if the bus time is less than posted time the bus model is executed. The behavior of the bus model is described in the Control of Events section above. The bus time is set equal to posted time after the bus model is executed.

Parallel Processing Environment

The computers used to run this simulation utilize parallel processors. To take advantage of this parallel environment, processes are partitioned among the processors according to task and event durations, parallelism in the real system, memory requirements and computational loads. Figure 9 illustrates one way this can be accomplished.

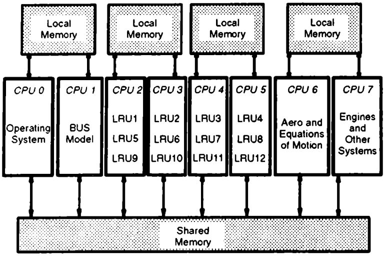


Figure 9 — Process Distribution vs Memory Usage

Distribution Based on Task and Event Durations

LRUs which have smaller frame times than other LRUs need to run more frequently. Bus transmitters typically transmit more frequently than the LRU tasks run. The high frequency tasks and transmitters are assigned to processes and processors separate from the lower frequency tasks to minimize idle time for the higher frequency tasks.

Distribution Based on Parallelism in The Real System

The system being modeled has a combination of LRUs which are separate computers running in parallel. An attempt is made to preserve this characteristic of the system. Each LRU is assigned to a separate process. The processes are grouped on processors such that they execute simultaneously as they would in the real system. The simultaneous running characteristic can not be completely preserved due to the limited number of simulation computer processors available. However, parallelism is still guaranteed by the design of the task and bus schedulers.

Memory Requirements Distribution

Each simulation computer processor has a fixed amount of associated local memory. When the processes assigned to a particular processor are so large that they exceed the associated local memory, global memory is assigned to accommodate the excess. When global memory is used up, the virtual memory mechanism will begin to swap to disk. The reversion to global memory and ultimately to disk have progressively more severe performance penalties. The processes are therefore distributed such that exceeding local memory is minimized.

Computational Loads Distribution

The aerodynamics, engines, gear and other aircraft systems which are not the main focus of this simulation are grouped together in relatively large tasks and thus require longer execution times than the typical LRU or bus model. LRUs require more execution time than the bus model. An attempt is made to group those processes with lower execution times on processors separate from the processes with higher execution times. Again the goal is to minimize idle time for the processes with the lower computational load.

Conclusions

This work is a significant advance in the technology of simulations for studying digital asynchronous flight control systems. It is being used to investigate and validate the design of a fly–by–wire flight control system for a modern transport airplane. The ability to alter the parameters which effect the asynchronous nature of the system allows the study of a wide range of possible variations in system characteristics.

This simulation capability can be used to study aspects of digital flight control systems that could not be done in traditional simulation environments or in bench studies of the hardware.

Acknowledgements

We wish to acknowledge members of the group which developed the software for this simulation. In addition to us, the members include Kipp Howard, Bruce Zunser, and Chris West.

We also wish to acknowledge the contribution of the flight controls engineers who supplied the requirements and model definitions for this simulation. Special thanks must go to Dana Olson for the basis of the bus model, to Ron Riter for his many valuable suggestions regarding the task schedulers, and Larry Hazard for his contributions to the design of the software.

AIAA-94-3414-CP

DEVELOPMENT OF THE DUTCH NATIONAL SIMULATION FACILITY NSF

-The world's premier motion-based F-16 MLU simulator-

by

H.A.J.M. Offerman*

National Aerospace Laboratory NLR

Amsterdam, The Netherlands

Abstract

The Dutch National Aerospace Laboratory NLR is currently developing one of the most advanced research flight simulators available today for fast jet aircraft, called the National Simulation Facility NSF. The NSF is planned to be a full scenario research flight simulator, in which highly realistic air-to-air and air-to-ground missions can be flown for evaluation of (new) tactics, training methodologies and very important, for assessment of necessary simulation and simulator requirements for specific training tasks. Design and development of the simulator is centered on the F-16 Mid-Life Update configuration. The paper will adress the various technical aspects of the National Simulation Facility.

1. Introduction

Research flight simulators are a valuable asset within the simulation arena. Consisting of modular components with the ability to represent in a flexible manner various types or configurations of vehicles to be tested, these research flight simulators are used to adress high-technology and early concept evaluation of virtually every flying vehicle.

The National Simulation Facility being developed at the Dutch National Aerospace Laboratory NLR situated in Amsterdam, can be regarded as one of the most advanced fast-jet research flight simulation sites becoming available in the near future.

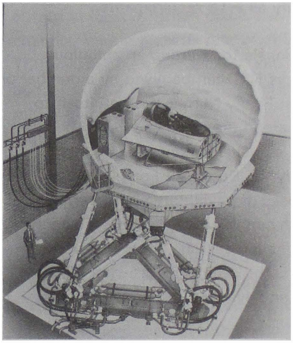


Figure 1. Artist Impression of the National Simulation Facility NSF

The NSF initiative initially was directed to be a separate extension of the existing NLR

* Project Manager National Simulation Facility
NLR Flight Simulation Department

Research Flight Simulator. In this simulator which is already operational since 1976, both research and development programs are conducted for national (e.g. Fokker, Civil Aviation Administration RLD) and international customers (e.g. Israeli Aircraft Industries, General Dynamics, Federal Aviation Administration). Only recently the simulation transport cockpit was modified to represent state-of-the-art glass cockpit technology for simulations under contract awarded by the US FAA, to assess the feasibility of ATC datalink communication. However, during the NSF feasibility study in 1991 and also later, when starting the development process early 1992, it became evident that next to procuring new simulator systems, a major upgrade for many existing systems deemed necessary to satisfy all F-16 requirements.

The rationale for starting an ambitious project as NSF can be given by a number of reasons: Firstly NLR committed itself to participate in the Lockheed F-16 Mid-Life Update co-development programme, secondly the Dutch Air Force indicated the usefulness of having a well-tailored F-16 research simulator for tactics evaluations and training research. Another important reason is the seemingly growing need for having an advanced research helicopter simulator.
These reasons have led to a phased development approach: during phase 1 the F-16 MLU simulation capability is developed, the NSF phase 2 development will concentrate on helicopter simulation. This phased design approach already proved to be worthwhile.

Partly financed by the ministries of Defence and Economic Affairs (the other part is coming from NLR's own research budget), the possibilities were present to create the high-technology facility in a relatively short time span.

2. NSF Requirements

The NSF initially was directed to be an extension of an already existing research flight simulation facility. Existing simulator components, both hardware and software were to be used as a baseline in the design of the NSF. These elements consisted of a host computer, electronic interfaces, a six degree of freedom motion platform, software development programs, real-time simulation executives and many other. Already during the feasibility study it became obvious that not all components could be used or upgraded with extra capability. Some of them had to change dramatically in which case a reassessment was needed, to be performed in the NSF design process.

The aim for NSF (phase 1) is to create a full mission, manned F-16 Mid-Life Update simulation capability for research and development. The top level requirements can be defined as:

- The facility should be capable to perform, on behalf of Lockheed's F-16 Mid Life Update codevelopment programme, the final functional cockpit mechanisation check before actual flight testing of the F-16 MLU will commence;
- Simulations should support air-to-air, air-to-surface, low altitude high speed missions with a maximum of realism;
- Simulator investigations concentrating on human factors are considered to be one of the prime issues of the facility's capabilities: research into training and

training effectiveness, assessment of pilot workload during simulated combat missions, etc.;

- Assessment of simulator training effectiveness and (training) simulator component requirements should be possible by selective 'degrading' of its components;
- Simulation of avionics should support integrating both aircraft equipment through its standard interface ("hot bench") and its functional software equivalent;
- The design and development should include the future growth to enable helicopter simulation;
- All hardware and software components should be designed in such a way that interchangeability with the other NLR research flight simulator is ensured.

Considering these requirements, which by no means are meant to be complete, one asks for a mere flight simulation centipede. With the usual available, abundant amount of budget to build such a facility and the fact that the Lockheed MLU codevelopment programme implied a date ready milestone, the project was (and still is) quite restricted.

Considering the essential specifications and technical requirements stemming from the above one could fill many pages of this paper. In the following paragraphs a limited set is mentioned, however these can be regarded as the most important:

2.1 Visual requirements:

- Air-to-surface missions and especially the low-altitude high-speed mission imply a highly realistic ground terrain to be portrayed to the pilot. The instantaneous field of view must be at least 120 degrees in azimuth, while the field of regard should not be restricted.
  Because of its very nature the research simulator should be able to investigate the effects of different sizes of field of view (FOV) on the training task. A requirement for the NSF visual system is that it must allow to expand the azimuth FOV to 180 degrees.
- A contrast ratio of 15:1 and a brightness of 6 ft-Lamberts minimum were required.
- The use of full colour fototexture is essential. Considering the future use of the NSF in performing mission rehearsal type simulations, large area geo-specific databases overlayed with phototexture imply the requirement of texture paging from disk.
- The image generator was required to output at least 1 million pixels per channel, with a pixel fill rate of at least 160 million pixels per second, with an update rate of 60 Hz. Quite different for e.g. a training simulator was the requirement to be able to reduce the IG transport delay from its nominal value of 80 milliseconds to 40 milliseconds.
- From an operational point of view, we strongly felt that the capability of wearing actual equipment like night vision goggles or helmet mounted displays should be a requirement for the simulator. Together with the required field of regard the requirement for the display system essentially was resticted to a dome system with some kind of area-of-interest projection.
- Performance of projection devices for the area-of-interest were required to meet eye accelerations and speeds to enable future use of variable acuity projection in combination with eye-tracking. The requirement stipulated in the contract

was, however, limited to head-tracking only; at that time we didn't feel that eye-tracking techniques had evolved to the extent necessary for the purpose.

- Because of the limited space in the simulator bay and the large excursions of the six degrees of freedom motion platform, the dome size could not be larger than 17 feet outside diameter. Before actually ordering the dome, the Dutch Institute of Perception (TNO-IZF) - now called the Institute of Human Factors (TNO-TM)- performed an investigation in the visual perception side-effects of such a small dome. Apart from the obvious serious misalignment of the actual Head-Up Display collimation distance with the real image on the dome, non were quoted as being considered potentially distractful to the pilot.
- Because of its very nature the NSF research simulator allows more than one cockpit to be used on the motion platform in conjunction with the same visual display system. It required the display system to be removable from the platform. Additionally, to make matters worse from the visual supplier's point of view an 25 Hz or above first eigenmode, 2 g requirement was imposed on the system.

2.2 Cockpit requirements:

During the NSF feasibility study it was pointed out that the facility should be multi functional in offering different sorts of cockpits and not restricting the facility to F-16 use only. Because of the financial implications, however, it was decided to postpone the development of the helicopter capability after finishing the phase 1, F-16 development. From a design standpoint however, the requirements for helicopter simulation were taken into account as much as possible.

The necessity of acquiring an exact replica cockpit of the Lockheed F-16 stemmed from one of the first projects to be carried out by the NSF: a complete functional checkout of the F-16 MLU cockpit mechanization, part of the Lockheed F-16 MLU codevelopment programme. Also the vast amount of human factors related experiments to be carried out in the NSF required the highest equipment fidelity attainable.

To be able to use different cockpits on the motion platform, specially constructed cockpit adapters are designed to locate the cockpit to the pilot's eye reference point.

2.3 Motion cueing requirements:

Requirements for motion cueing can be regarded as an important issue in research flight simulation. While for training simulators, specially in the fast jet area no platform motion systems are claimed to be necessary, one of the important NSF objectives is to define these training system requirements. By combining both high-performance visual and platform motion systems, the NSF is capable of establishing objective and subjective criteria for the future fast jet training simulator.

In the recent past Israeli Aircraft Industries developed with NLR the digital flight control laws for its LAVI aircraft on the Research Flight Simulator. All simulation experiments were carried out with platform motion to refine the control law filters and transitions for optimal handling of the aircraft during all flight modes. The results from these experiments (and many other) lead to the certain belief that high-fidelity simulation

needs platform motion, also for fast jet purposes.
The six degrees of freedom synergistic motion system which was built by the Dutch firm Hydraudyne on NLR specifications is designed for simulation of fast jet and helicopter manoeuvres. Table 1 lists the most important specifications of this system.

Complementary to platform motion cues are those which will be generated by a G-cueing system. The system requirements demand the simultaneous interaction of:

- a G-seat;
- an anti-G-suit;
- lapbelt and shoulderharness loaders;
- a helmet loader and
- a partial positive pressure breathing system.

Time delays of the G-seat were to be kept to a minimum (at least below 70 milliseconds) which will require electrical instead of pneumatic actuation.

2.4 Simulation model requirements:
Quite different from the traditional requirements imposed on research simulation is that the NSF would be required to have a full avionic suite and system simulation capability next to its normal high-fidelity aerodynamic and flight control system models. Being not able to develop all the F-16 Mid-Life Update avionic models in-house within the short time span available, Lockheed Fort Worth Company supplied from their Flight Simulation Laboratory a set of models. These models consist not of the actual airborne software modules, but simulate the F-16 avionics functionally.

The functional requirements applicable for the simulation program on a higher level, dictated the need for exchanging avionic software modules with their equivalent hardware box. This implied that communication and interface requirements for a large number software models were to be based on the MIL-STD-1553B protocols and bus structure. We clearly felt this to be important to allow for evaluation of aircraft system conceptual design, development and integration. This capability has been used on the civil research flight simulator for many years with an ARINC-429 bus structure. This capability has been used for several Fokker 100 EFIS display certification purposes,

2.5 Networking requirements:
An important requirement for almost every modern (military) simulator is to be able to communicate with other simulators over a real-time digital network. Also the NSF will use the capabilities available in the 'de-facto' standard IEEE-Std-1278-1993 "Protocols for Distributed Interactive Simulation (DIS)" for these applications. NLR already demonstrated its DIS compatibility during the 1994 International Training Exhibition and Conference (ITEC) in The Hague, Netherlands late April. In a scenario (with some 12 other players) reflecting an armed United Nations operation, NLR contributed with an F-16 performing an air-to-surface attack. It was flown from the F-16 mock-up situated at NLR in Amsterdam, some 30 miles from the Conference Center. Two way DIS communication with the Conference Center was made through an ISDN connection.

2.6 Software and computing requirements:
While considering the computing requirements based on the F-16 MLU avionics simulation software running at 50 Hz, it became obvious that a mere processor upgrade of the existing Concurrent Computer Corporation MicroFive host would be inefficient. Functional requirements indicated that next to raw processing power a clear need had arisen for using UNIX and a more user friendly (graphical oriented) software development environment.

These functional requirements were put out as a Request for Proposal (RFP) to various suppliers of real time computers. After a carefull selection based on the Multi-Element Component Comparison (MECCA) method, the Silicon Graphics CHALLENGE computer came out as favourite. We decided to procure two CHALLENGEs, one dedicated for real-time simulation, the other dedicated for software development. In this way, the software modellers are not bothered in their work when real-time simulations take place. The graphical user interface could be implemented by using X-terminals.

2.4 Interfacing requirements:
Interfacing the host computer (where all the data is generated) with the other simulator components like cockpit, visual system etc., can be regarded as a difficult subject. In the NSF project we decided to make use of the latest technology available in this field; high-speed digital fiber-optic connections.

Before the NSF started in 1992, a new electronic interface system for the Research Flight Simulator was being developed. It was based on the philosophy of distributed processing for each (major) simulator hardware system which would be coupled through a fiberoptic ring network. This philosophy would relief the host computer from doing non-relevant computations; e.g. dedicated hardware driver software would be swapped to nodes (basically front-end computers connected to the hardware). All nodes would be linked in ring-like fashion, and every node would take only data from the link when needed or put out data when requested.

Because for the Research Flight Simulator such a custom-made electronic interface system had been developed, it could for the greater part be reused for the NSF, only parts of the architecture were based on the proprietary Concurrent internal computer bus. The largest part of the basic design is based on commercially available VME boards and operating systems.

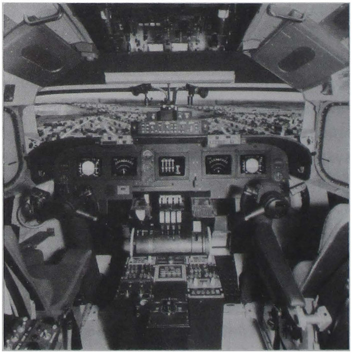


Figure 2. Flight deck of the civil oriented Research Flight Simulator transport cockpit

3. Development

In the following sections the NSF software and hardware development process will be explained in more detail. The paper will touch only briefly on the development time schedule.

3.1 Schedule:

The main schedule for the NSF development was fixed right from the start through the milestone dictated by the Lockheed codevelopment programme. This requires NSF to become operational during the final quarter of 1994. Also, during the entire 1995 time frame, the NSF is scheduled for operational use in two Dutch National Technology Projects, directed by the Dutch Air Force, as well as performing experiments for the EUCLID RTP 11.2 programme (described in more detail in section 5).

3.2 Software development:

The above mentioned requirements had to be translated to technical specifications and strict project development rules. While not enforced by a military body to adhere to the MIL-STD-2167 software development standard, the NSF project management decided to work with a more pragmatic, but still sufficiently documented and controlled software development environment.
Apart from defining the top level user requirements and system interface design, no subsystem user requirements were thought to be needed (most subsystems would be re-used anyway). The subsystem design proces is stipulated by:

1. Functional Requirements or Functional Design
2. Technical Design
3. Detailed Design and Implementation (coding)

where both functional and technical design phases are concluded by means of a formal review. Together with strict interface configuration management, the NSF project management can take things easy.

Software development, at least the design proces, is based on the Yourdon Structured Analysis for Real Time systems (SA/RT). The software package TEAMWORK supports the SA/RT method and thus much time consuming checking of interface definitions and dataflows is automated.

While every project has its share of pitfalls, the NSF project seriously underestimated in the beginning of the programme in 1992 the amount of software development to be needed. The main reason is simple: we did not anticipate to start from scratch. This needs a little explanation: again, the NSF was meant to be an extension of the existing Research Flight Simulator. Of its own it had a wealth of real-time aircraft models, real-time executives, computer aided software engineering (CASE) tools and many more. Most of the FORTRAN based CASE tools were already a decade in operation. Although continuously being improved and extended and having proved themselves during the many succesful simulator programs, it was thought that:
- a major upgrade had to be implemented in the NSF development or
- be postponed for another decade.

We choose for the first option: the entire simulation program, from real-time executives to the aircraft model developer tools were to be renewed. This implied that a full year was needed to reassess the present capabilities and extend them with latest technology possibilities in software development.

In close cooperation with two Dutch subcontractors, Fokker Space & Systems and BSO, early in 1994 NLR came out with a new Real Time simulation System and Software Development System. Both packages are becoming now the "de-facto" standard within NLR (and possibly outside NLR as well).

3.3 Hardware development:

In this section the different NSF hardware components are described.

The visual system for which a request for proposals was sent beginning 1993, was decided to be a system from Evans & Sutherland. The combination of a two-channel ESIG-3000® AT/GT and the VistaView® head-slaved projection system met all the NSF requirements.

The VistaView® projection system displays the image from the IG on a 16.6 feet graphite composite dome, with both a head-slaved background of 120 x 90 degrees and inset of 40 x 30 degrees FOV. The ESIG-3000® AT/GT combines all the latest technology gadgets available within one system, although no revolutionary breakthroughs were to be expected. The two channel system is configured to handle 1000 polygons in the inset and 3000 polygons in the background, thus giving a balanced polygon density in inset and background. Quite unique is the capability that we can modify the IG update rate and distribution of polygons over inset and background by changing the hardware configuration.

Texture paging allows texture maps to be swapped from disk during database traversing. The "Global Texture GT" ESIG-3000® option included in the system allows the IG to page separately from disk (through separate buses) polygonal data and texture data. This provides maximum throughput efficiency and unburdens the front-end processor and graphical pipeline. More time can be spent on e.g. moving model calculations.

The VistaView® projection system optically combines the image coming from two Martin Marietta light valves into one focal plane. Both the images for background and inset are pre-warped in the IG, where also a non-linear image mapping function corrects for not having the exit pupil of the optical system in the center of the dome. After de-rotation the one image is projected by means of a servo-controlled lens. Capability of the servo system is better than 4000 deg/sec^2 and 1000 deg/sec, which for future growth allow the servos to track the eye line-of-sight for variable acuity IG techniques.

The requirement of changing the field of view is implemented by changing an optical element in the VistaView® and reconfiguring the IG through its configuration file. In this way the FOV can be changed from the nominal 120 degrees in azimuth to 180 degrees. The elevation FOV changes accordingly.

The dome has a high-gain coating (gain 5) which was felt as a good compromise between brightness fall-off and the available brightness at the pilot's eye point.

The visual system was ordered with a Database Modelling Station, the Evans & Sutherland EaSIEST® system. With this system we can modify and create visual databases. By scanning satellite photos, geo-specific texture maps can be created which are correlated with DMA DTED and DFAD data. Notwithstanding the fact that the visual modellers probably will need some time to get acquainted with the system, the possibilities for mission rehearsal type applications are there.

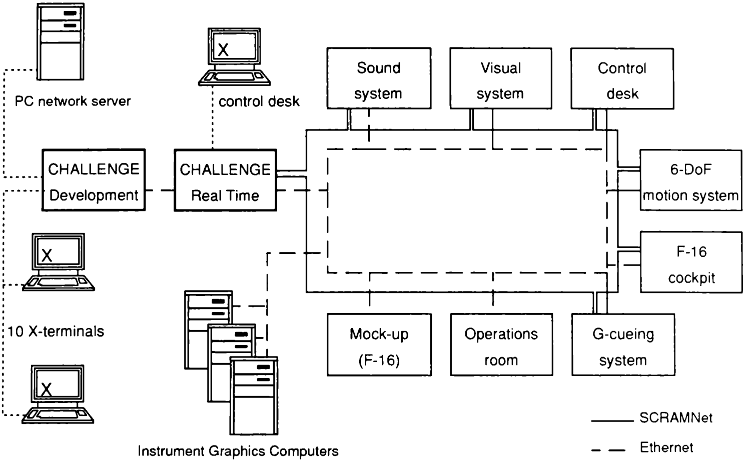


Figure 3. NSF Flight Simulator Interface System

Development for the NSF interface system could for the greater part be continued on work already performed for the Research Flight Simulator.

Decided when the SGI CHALLENGE host computers were selected, it was required that the external real-time interfacing would be based on the Systran SCRAMNet system. Each NSF simulator subsystem which will need data from the host computer has a connection on the fiber-optic link with a SCRAMNet card (see figure xx). From there on each node will proces whatever is needed. For instance the F-16 cockpit interface node has additionally to the SCRAMNet card and VME Motorola-based processors, some 40 cockpit analog inputs, 30 cockpit analog outputs, 144 discrete inputs, 272 discrete outputs, two MIL-STD-1553B buses and about 40 various other signals to be processed (on VME boards).

For non real-time interfacing, for instance for loading the subsystem configuration files before starting the real time proces, use is made of standard Ethernet communication.

The F-16 cockpit was procured from Lockheed Fort Worth Company through the Royal Netherlands Air Force (RNLAF). When we started the NSF project, it was still not decided by the Dutch government whether they would sanction the Mid-Life Update (MLU) of their F-16 fleet. Unfortunately, this was still the case when we decided to order an F-16 cockpit in MLU configuration. But we had to, because of the lead times necessary for building the cockpit and ordering MLU specific equipment. Everything worked out well, fortunately. The NSF F-16 MLU cockpit consists of a real aircraft section, adapted by Lockheed for simulation purposes and placement on a motion platform. Most of the instruments, displays and controls are exactly as in the aircraft, however some specific instruments (like altimeter) had to be exchanged by special simulator instruments.

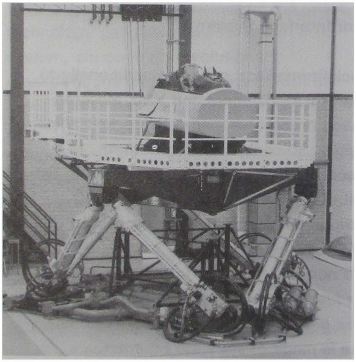


Figure 4. F-16 MLU cockpit on the 6-DoF motion platform before dome integration

Apart from a dedicated F-16 cockpit also a wooden F-16 mockup has been developed. With the physical dimensions and distances to controls and displays as in the real aircraft, this mockup will be used for evaluating experiment feasibility and to test early concepts for human factors research. Within the NSF project we use the mockup to test the avionics software on its functionality. The mockup is equiped with three colour monitors overlayed with touchscreens and one 19" monitor for representation of Head-UP Display and limited out-of-the-window view. We simulate the HUD and the two Multi Function Displays (MFDs) through our Silicon Graphics IRISs, the MFD's bezel switches can be interfaced by the pilot through the touchscreens. On the third display also other flight instruments like altimeter, HSI, angle-of-attack indicator etc. can be portrayed.
Because the mockup makes use of exactly the same software as the F-16 simulator, it can be used as a manned wing man or as a threat in the simulation scenario.

The sound generation system is a digital system based on the Paradigm Emax sound synthesizer and state of the art digital sound processors integrated by the Dutch firm CHeSS. Together with the Dutch Institute for Human Factors, we realized true F-16 sounds in the system by recording in a flying F-16, with a microphone in the pilot's ear the aerodynamic hiss and engine sounds for a large number of conditions. The sound system is prepared to handle three dimensional sounds, which will possibly be a next step in lowering pilot workload and improving situational awareness in the aircraft.

The G-cueing system is now the only part in the NSF development which had to be postponed and thus will not be available at the end of 1994. Probable integration will take place in the end of 1995.

An Operations room will be available as soon as the flight simulator building is extended with an additional floor. However, for the time being, the operations room will be functionally implemented in the NSF control room. From various monitors the tactical scenario can be viewed as well as most of the cockpit instruments. An experiment and scenario manager will control the content of the simulations. For more training oriented simulations, this operations room additionally functions as the Instructor Facility.

4. Future developments

The National Simulation Facility will continuously be upgraded to reflect state of the art technology in flight simulation. To mention the most important developments which will take place in the coming years:

- Development of a manned helicopter simulation capability, the NSF phase 2. The recently created Dutch Light Mobile Brigade is being equiped with large numbers of helicopters of various type: large and medium transport and armed helicopters. To support the Air Force operating these helicopters, the NSF will be adapted to also simulate some of these.

  Development of the various helicopter simulation models already is taking place. Calculations based on the advanced blade-element method for main rotor simulation, amongst others, guarantees high fidelity in helicopter simulation.
- The visual system will be extended to allow for radar image generation. Based on the same database the generation of a correlated radar ground map is considered to be essential in most military tasks.

5. Future Projects

The National Simulation Facility is already scheduled to take part in a number of simulation projects.

- Already mentioned a few times is the participation in the Lockheed F-16 MLU codevelopment program. The NSF will perform one out of two final cockpit mechanization checkouts, the other taking place at Lockheeds facility in Fort Worth. In the checkout simulations, pilots from five nations (USA, Belgium, Norway, Denmark and The Netherlands) will evaluate if all the changes made in the avionic functionality are implemented correctly and whether the avionics meet all the specifications. Approval from this Combined Cockpit Review Team is essential for Lockheed to proceed with finalizing the on-board software and to start with flight testing.

  This project will be the "proof of the NSF pudding", and will indicate whether the three years of hard work can do the job!
- In the European programme of EUCLID (EUropean Cooperation for the Long Term in Defence, a programme to push research and development and to share results between participating nations) a number of simulation projects have been defined. From research into establishing training requirements and needs to full scale avionic development.

  The NSF will take part in EUCLID Research Technology Project (RTP) 11.1: "Human factors" and in RTP 11.2:

"Simulation Technology". In each case the NSF will be used for correlating results with tests performed in the actual aircraft.

- The Dutch Air Force (RNLAF) started two National Technology Projects in which the NSF will have a crucial role. One project will investigate the use of voice recognition and voice commands in the different F-16 tasks, the other will try to identify pilot workload depending on the task performed and correlate results with the actual aircraft.
- Not yet contracted but indicative for the possibilities is the integration of an aircraft Helmet Mounted Display in the NSF. The HMD lends itself for off-boresight aiming and targeting of aircraft sensors which can be demonstrated in the simulator in various tactical and weather environments under controlled conditions. An important issue is that the simulator is used to verify software and hardware functionality before integration with aircraft takes place.
- Combining the possibilities created for our civil customers in the field of Micro Wave Landing system evaluations and the F-16 flying characteristics, is to evaluate the implementation of MLS on NATO airbases or for out-of-area operations. The NSF can be used to evaluate the various approach and departure possibilities for different airfields and situations. E.g. optimal paths can be defined for noise abatement.

6. Concluding remarks

Development of the National Simulation Facility phase 1 is currently nearing its end. Final integration and testing will take place by the end of 1994. By then the research facility will be unique in offering capabilities such as high-performance motion and visual cueing together with a full mission environment. The NSF will be the world's first motion-based F-16 MLU simulator. The facility will not only offer extensive research and development possibilities for both Air Forces and industry, but from an operational point of view, it will also be an asset in defining training (simulator) requirements.

During the 1995 timeframe, the NSF will be extended to allow for advanced helicopter simulations. Based on high-fidelity simulation models, the intrinsic value of the simulator will increase dramatically. By virtue of designed and incorporated flexibility and modularity, the National Simulation Facility presents itself as an international testbed for advanced simulation possibilities.

Table 1: Six degrees-of-freedom (Hydraudyne) high-performance motion platform system:

	displacement		velocity	acceleration
	pos	neg	(pos/neg)	(pos/neg)
longitudinal	1.72	1.34 m	0.8 m/s	8 m/s^2
lateral	1.39	1.39 m	0.8 m/s	8 m/s^2
vertical	1.01	1.14 m	0.8 m/s	10 m/s^2
roll	30	30 deg	30 deg/s	200 deg/s^2
pitch	29	29 deg	30 deg/s	200 deg/s^2
yaw	40	40 deg	30 deg/s	150 deg/s^2

fully hydrostatic actuators
band width: 45 degrees phase lag at 4 Hz

AIAA-94-3415-CP

MODERN FLIGHT SIMULATORS FOR RESEARCH APPLICATIONS

Prof. Dr.-Ing. Gerhard E. Huettig *
Berlin University of Technology, Institute of Aeronautics and Astronautics
and
ZFB Zentrum für Flugsimulation Berlin GmbH
10587 Berlin, Germany

and

Dr.-Ing. Herbert F. Bernard **
CityLine Simulator and Training GmbH Berlin
and
ZFB Zentrum für Flugsimulation Berlin GmbH
10587 Berlin, Germany

Abstract

Todays research activities in the fields of aeronautics require sophisticated tools to verify and validate new developments. Such a modern testbed is available at the Institute of Aeronautics and Astronautics at the Berlin University of Technology with an Airbus A340 Full Flight Simulator (FFS). This A340 FFS operated by ZFB Zentrum für Flugsimulation Berlin is enhanced with a Scientific Research Facility (SRF). The unique simulator configuration of having a second host computer for research purposes provides a flexible use of the simulator for crew training as well as research applications. The research capabilities of the simulator and some of the research projects underway and proposed are presented.

Introduction

Increasing air traffic, limited airspace, new generation aircrafts with glass cockpits and highly automated aircraft systems characterize the situation in todays and tomorrows air traffic.

Research activities in the various fields of aeronautics require modern and realistic tools to verify and validate new developments. The use of real aircrafts as testbeds in a real traffic environment is for most research activities next to impossible.

Full flight simulators with a sophisticated visual system provide a modern and realistic testbed.

The costs for such a modern flight simulator, however, are very high and most institutions cannot afford to buy an expensive flight simulator for research purposes only.

ZFB Zentrum für Flugsimulation Berlin operating an Airbus A340 Full Flight Simulator (FFS) found a way to use the simulator for pilot training as well as for research purposes.

System Architecture

A full flight simulator with a visual system simplified consists of:

- the host computer, controlling all simulation processes,
- a simulator interface which drives the controls and indicators in the cockpit and which contains original aircraft avionics, simulator power supplies, etc.,
  the cockpit with motion platform and visual system, and

* Professor, Aeronautical Engineering

** Aerospace Engineer, Member AIAA

a visual computer generating the proper visual image for each flight situation.

The host computer software is usually approved for pilot training by the national authorities. This fact prohibits any modification of the software modules for research purposes.

In order to enable software modifications ZFB has enhanced its simulator with a Scientific Research Facility (Figure 1). Heart of the Scientific Research Facility (SRF) is the main research computer which can be used instead of the training host computer. Switching over to the research computer allows software modifications without touching the licensed training software.

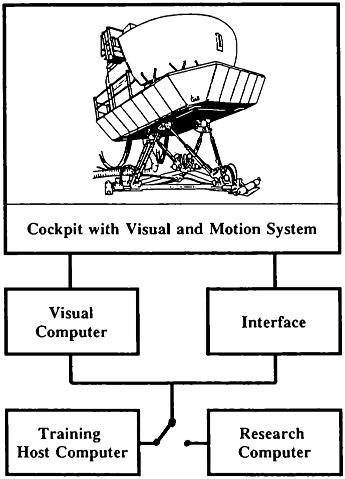


Figure 1: Block Diagram of ZFB´s A340 Full Flight Simulator with Research Facility

Further SRF components are:

a Display Development Computer, which allows the creation of new display images,
an Avionic Test Bench (Figure 2) for integration of user-designed or alternate avionic hardware into the simulation process,
an Audio and Video Recording Facility.

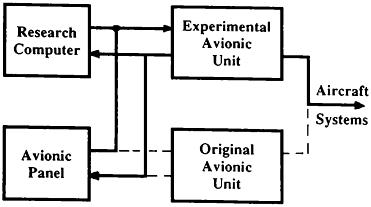


Figure 2: Test Bench to integrate Experimental Aircraft Avionic into the simulation process

System Configuration

The simulator allows two basic modes of operation: Training and Research.

In RESEARCH mode the flight compartment is connected to the research host computer allowing software modifications and various other research activities.

In TRAINING mode the simulator operates like every other simulator. The flight compartment is connected to the training host computer using the certified training software.

A switchover from one mode to the other can be accomplished within 10 minutes.

In order to enable research activities, even if the flight compartment is not available, the SRF is designed to be operated also as a stand-alone facility. This is important since the access time for research activities is not limited by crew training, and research programs normally require the flight compartment only in their final stage.

The stand-alone capability of the Scientific Research Facility could be achieved by integrating additional software which simulates the aircraft avionics in the simulator interface which is not available for research when the simulator operates in training mode, e. g. autopilot, electronic flight control system and flight management and guidance system.

Research Capabilities

The SRF provides a variety of research features:

Data Recording

The SRF allows data recording of up to 400 parameters sampled at 60 Hz. In addition to these simulator parameters, 20 pilot parameters, such as heart beat and blood pressure levels may be recorded from the cockpit.

Software Modification

Modifying the software simulation modules enables a wide variety of research possibilities. Flying the A340 with experimental engines through an area with severe windshear and landing it with a newly designed landing gear is just one conceivable scenario.

Avionic Simulation

The availability of software simulation of the aircraft avionics, e.g. Electronic Flight Control System (EFCS), Flight Management System (FMS), Flight Guidance System (FGS) and Flight Envelope System (FES) is necessary for the SRF stand-alone operation and enables modification of various avionics functions.

Display Development

Using TIGERS (The Interactive Graphics Environment for Real-time Systems) developed by CAE Electronics, the SRF enables rapid prototyping of highly dynamical graphic displays.

The use of free programmable display units which can replace the original aircraft display units, e. g. Primary Flight Display (PFD), Navigation Display (ND) or the displays of the Electronic Centralized Aircraft Monitoring System (ECAM) allows the researcher to integrate newly developed display symbologies into the simulator cockpit.

The integration of novel displays is not limited to symbologies developed on the SRF workstation. Integration of displays designed on other workstations like the IRIS from Silicon Graphics is also possible.

Video Recording

Three video recorders present in the SRF are able to record the pilot´s and copilot´s actions by means of two in the flight compartment installable video cameras. The third video channel for example can be used to record the touch screen of the flight instructor or to record the displayed information, when using the free programmable display units.

Audio Recording

The video recorders also allow a recording of up to six channels of the ambient cockpit sound and the communication between pilot, controller and instructor.

Reconfigurable Test Bench

By integration of user-designed or alternate avionic hardware into the simulation process the researcher is able to stimulate these avionic boxes and to examine the input/output behaviour.

Present and Proposed Research Applications

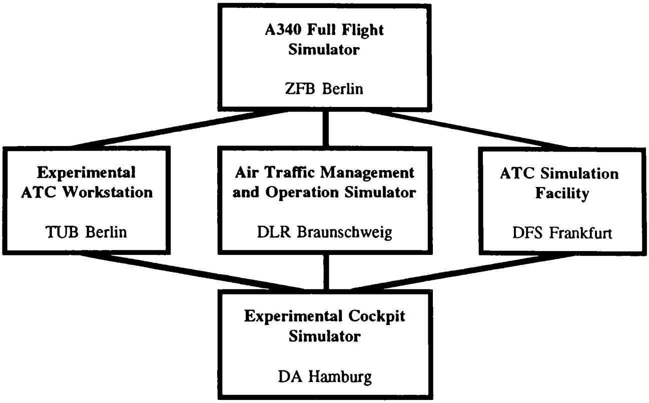
Research projects underway and proposed at the present time include:

Air Traffic Management

In the frame of a future Air Traffic Management research project it is planned to link ZFB´s A340 simulator together with other simulators in Germany to Air Traffic Control (ATC) simulators in German (and later European) research organizations (Figure 3).

It is also intended to link an experimental ATC workstation, developed by the Berlin University of Technology, to the A340 simulator.

These research scenarios allow the simulator(s) to fly in a simulated ATC environment.


A340 Full Flight Simulator
ZFB Berlin
Experimental ATC Workstation
TUB Berlin
Air Traffic Management and Operation Simulator
DLR Braunschweig
ATC Simulation Facility
DFS Frankfurt
Experimental Cockpit Simulator
DA Hamburg

DA	Deutsche Airbus GmbH
DFS	Deutsche Flugsicherung GmbH
DLR	Deutsche Forschungsanstalt für Luft- und Raumfahrt e. V.
TUB	Technische Universität Berlin
ZFB	Zentrum für Flugsimulation Berlin GmbH

Figure 3: Link of Flight Simulators and ATC Simulators to perform Air Traffic Management studies

FMS Operation and Control

In a study about the operation and control of the Flight Management System (FMS) it will be investigated how often the available FMS functions are used in the different phases of flight.

Systems Control by Touch Input

This project investigates the advantages and disadvantages of using touch screen devices instead of conventional keyboards for systems control. One application might be the operation of the Flight Management System.

Air Traffic Control Information Displays

One approach to increase the capacity of Air Traffic Control (ATC) systems is the implementation of an air-ground data link providing the Air Traffic Management computer on the ground with all related on-board data but also transmitting ATC messages into the cockpit.

This study addresses in particular the development and integration of a display into an advanced glass cockpit, which presents tactical ATC messages. The procedures to accept the incoming ATC messages and the consequences on the visual channel and mental workload of pilots are investigated.

Instead of installing an additional display it is preferred to use a part of the existing Navigation Display to present the ATC messages.

Perception of colored digital displays in aircraft cockpits

Electronic Instrument Systems in modern glass cockpits provide a great number of alphanumeric and graphic information in different colours on a small display.

Various experiments simulating color weakness as well as long- and short-sightedness of the pilot investigate pilot´s perception of the aircraft displays.

Taxi Guidance Systems for Future Aircrafts

This project addresses the possibility of substituting the information on the Captain´s Navigation Display (ND) by Taxi Guidance information when the aircraft is on ground. Operational advantages and the proper symbology are investigated.

Conclusion

Full Flight Simulators provide a sophisticated tool to verify and validate new developments in aeronautics. ZFB´s unique simulator configuration of having a second host computer for research applications allows a flexible use of the simulator for crew training as well as research purposes.

About the authors

Gerhard Huettig is presently a Full Professor and Head of the Section Flight Guidance and Control/Air Transportation at the Berlin University of Technology and acts as Managing Director of ZFB Zentrum für Flugsimulation Berlin GmbH. He earned a Ph.D. in aeronautical engineering and held management positions with Airbus Industrie and Lufthansa. He is also a current licensed airline pilot on A320.

Herbert Bernard is systems engineer for flight simulation and coordinates the research activities on ZFB´s Scientific Research Facility. He also works for CityLine Simulator and Training GmbH operating a Canadair Regional Jet FFS. He earned a Ph.D. in aeronautical engineering, worked as development engineer and teacher for aeronautics, and held a research associateship from the National Research Council (NRC) in Washington. During this time he worked for NASA Ames in the human factors department. Herbert Bernard is member of the AIAA.

AIAA-94-3416-CP

THE 'SMART SOFTWARE - SIMPLE HARDWARE' CONCEPT FOR MAXIMUM FLEXIBILITY IN RESEARCH FLIGHT SIMULATION

J.M.Hoekstra
Flight Simulation Department
National Aerospace Laboratory NLR
Amsterdam, The Netherlands

Abstract

Flight simulation can be divided in two main areas: flight simulation for training air crews and flight simulation for research purposes. In this paper an overview of research flight simulation specific technology is illustrated by some examples from the Research Flight Simulator (RFS) of the National Aerospace Laboratory (NLR). Then from these examples a common characteristic feature is presented which can serve as a guideline for research flight simulation technology.

RESEARCH VS TRAINING FLIGHT SIMULATION

Below (figure 1 and 2) two simulator cockpits are shown, representing the two main areas where flight simulation is exploited: research and training. Most other uses of flight simulation can be placed under one of these two headers. The military mission rehearsel for example is a form of training, while some tactical evaluations can be regarded as research. A third category form simulators built for entertainment. Because the technology of these simulators is still very different, this category will not be discussed here.

Though the two simulator cockpits in the figure resemble each other, the technology of research simulators and training simulators is quite different. This is due to the different purpose of these simulators, leading to different requirements.

WHY FLEXIBILITY?

Table 1 shows some requirements that are specific for research simulators and training simulators. In this paper the specific technology for a research simulator will be discussed. Typically a research simulator cockpit is constantly rebuilt and modified

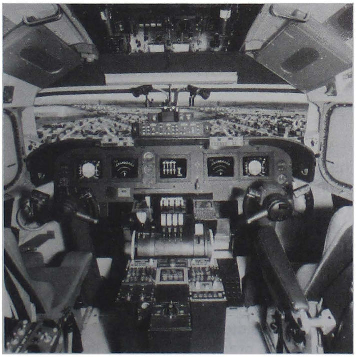


Fig. 1 Research flight simulator cockpit

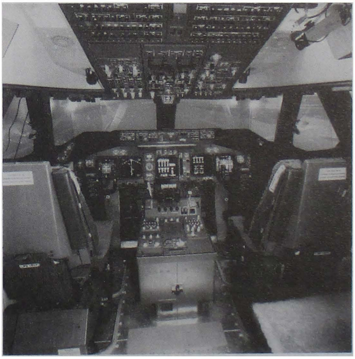


Fig. 2 Training flight simulator cockpit

TRAINING SIMULATORS:	RESEARCH SIMULATORS:
- degree of realism depending on nature of training	- degree of realism depending on nature of research
- failures should be simulated realistically	- different aircraft types have to be simulated
- hardware & software represent one specific aircraft type	- experiment scenario's have to be simulated realistically
- user friendly instructor-station	- hardware & software should be easy to modify
- should comply with regulations	- extensive real-time data recording
	- ....? (any experiment dependent requirement)

Table 1 Requirements for research and flight simulators

several times a year. To minimize this effort, a high degree of flexibility is essential for a research flight simulator. One should be able to modify, replace, expand and adjust all systems in the cockpit and often even be able to simulate different aircraft types.

Furthermore, research flight simulators require not only 'basic' flight simulation but also special features, like extensive data recording, even sometimes physiological measurements.

It is difficult to specify how realistic the simulation for an experiment should be, so a researcher often wants to be on the safe side and wants as much realism as he can get. This requirement for realism conflicts with the requirement for flexibility.
As an example of this for simulating analog flight instruments the most flexible, less realistic would be to draw the instruments on the screen of a graphics workstation, while the most realistic option, building the real instruments, is not very flexible.

When solving this contradiction you easily run into the well known simulation vs. stimulation dilemma. Should we take the real system and fool it with simulated inputs (the stimulation option) or should we try to replicate the system adapted for the simulator (the simulation option)? To complicate things in a research simulator you want to be able to do both. Sometimes an easy-to-modify simulation of a system is used for conceptual design evaluation and later the real system is 'stimulated' for final evaluation or certification.

Another feature of research simulators rarely found in training simulators is the ability to simulate different aircraft types. This is a second kind of flexibility that is required for research flight simulators. The NLR RFS is used to simulate Boeing 747-400/200, Fokker 100, Cessna Citation business jet, Swearingen Metro turboprop, a helicopter model and, with a fighter cockpit, the F-16.

How a high degree of flexibility is reached in the NLR RFS transport cockpit without sacrificing the realism, is what you will see in some examples before defining a common concept or guide line for research vehicle simulation technology.

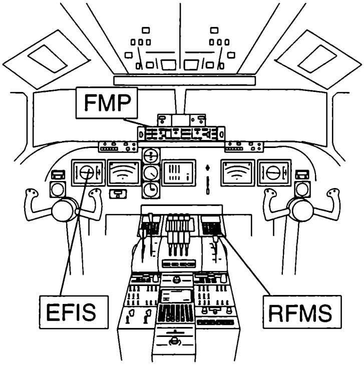


Fig. 3 NLR Research Flight Simulator cockpit

EFIS

The EFIS in the RFS is generated by graphical workstations. Up to four displays are drawn on the screen of a workstation. A device, called videosplitter, converts the screen picture to four seperate video signals, dividing the screen in four quadrants. These pictures are displayed on seperate tubes in the cockpit. The program drawing the pictures is written by the NLR and uses the graphical library of the workstation. This program has access to all variables of the main simulation program. This yields a high versatility enabling the use of totally new displays with new symbology and new information, like enhanced vision (using IR/radar as sensors) or a tunnel-in-the-sky display.

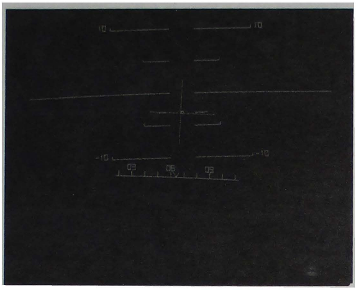


Fig. 4 Head-Up display of NLR RFS

A Head-Up-Display (HUD) is generated in the same way. The HUD is drawn on a graphical workstation. The resulting HUD-video signal is mixed with the out-of-the window view video signal, producing a good simulation of an ideal wide-angle HUD[2].

Next to this flexible configuration, there is the alternative of using the actual EFIS displays, with the ARINC-interface in the simulator, for evaluation/certification of real flight hardware. The Fokker 100 EFIS symbology was evaluated and certificated using this approach[3,4].

An interesting feature for display designers is the compatibility of the RFS with the DDF/NADDES display format[6]. This generic display format allows a display designer to specify and test a display in DDF format on eg. a PC on his desk. The resulting DDF-description of the display can directly be used in the simulator without any modification. This smooths the design traject enormously and thus enables the testing of display concepts in the realistic environment of the simulator.

FLIGHT MODE PANEL

The flight mode panel (or mode control panel) is used to control the autopilot and autothrottle functions. In the NLR RFS this panel is basically a panel with switches, dials, lights and digital displays. The logic of arming and engaging modes is very

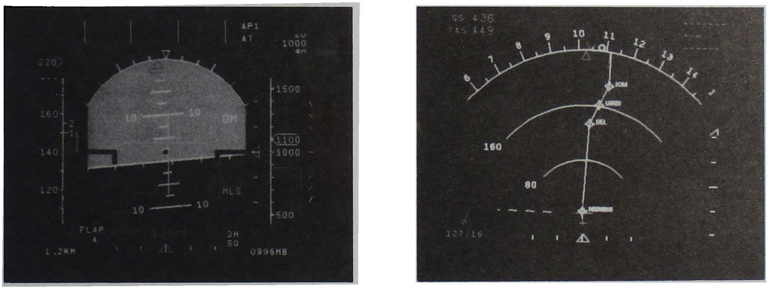
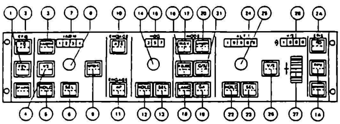


Fig. 5 Fokker Electronic Flight Instruments System (EFIS)

In the figure above left the primary flight display and right the navigation display in map mode are shown. This EFIS lay-out is based on the Fokker 100 format. The Fokker 100 EFIS symbology was evaluated and certificated using the research flight simulators of Fokker and NLR in the period from 1981 to 1987. Several changes resulted from the experiments. The final result is a very neat EFIS picture. This EFIS symbology is used for several aircraft types in the NLR RFS. Pilots are always very pleased with this format, which is easy to learn and takes a very short time to adapt to. Sometimes for project dependent reasons however the EFIS symbology is changed to the actual format of the type of aircraft to be simulated. Due to the Fokker 100 project the EFIS program is easy to adapt to all user specified formats. In the certification phase of the Fokker 100, actual EFIS tubes were used in conjunction with the ARINC Bus Interface System of NLR's research simulator.



1	FLIGHT PATH VECTOR
1a	FLIGHT PATH VECTOR
2	FLIGHT DIRECTOR
2a	FLIGHT DIRECTOR
3	IAS/MACH SELECTOR
4	V2 SELECT ON/OFF
5	SPEED HOLD
6	SPEED SELECT
7	SPEED DISPLAY
8	SPEED KNOB
9	SPEED ON ELEVATOR
10	AUTOTHROTTLE ON/OFF
11	AUTOPILOT ON/OFF
12	HEADING HOLD
13	HEADING SELECT
14	HEADING KNOB
15	HEADING DISPLAY
16	VOR/LOC SELECT
17	LNAV SELECT (FMS)
18	LAND MODE
19	GO AROUND
20	VNAV SELECT (FMS)
21	GLIDE SLOPE
22	ALTITUDE HOLD
23	ALTITUDE SELECT
24	ALTITUDE DISPLAY
25	ALTITUDE KNOB
26	VERTICAL SPEED SELECT
27	VERTICAL SPEED WHEEL
28	VERTICAL SPEED DISPLAY

Fig. 6 Flight Mode Panel

complex and dealt with by software. But simple logics as the increase of the display value due to clicks of the dial is also performed by the software. So this is a good example where all the intelligence is in the software resulting in very simple and straightforward hardware and maximum flexibility. It can easily be adpated for different aircraft types.

FLIGHT MANAGEMENT SYSTEM (FMS)

Today's aircraft are equiped with a flight management system (FMS) for navigation, performance management and cost management. This 'brain' of the cockpit can be divided into mainly two systems: the Command & Display Units (CDU's) and the Flight Management Computers (FMC's). The CDU is the man-machine interface and is used by the pilot to enter data like route changes. The FMC is the main computer of the system performing all the calculations and autopilot guidance commands. Both contain some logic and communicate with each other. In the NLR RFS again the logic is in the software running on a workstation. This program contains both the FMC- and CDU-simulation. The contents of the CDU screens is drawn on a graphical workstation and sent to the CDU screens via the video splitter. The program reads the codes of the keys pressed on each CDU. Everything that happens in-between is dealt with in the simulation program, which is written in-house, based on information from the operations manual. The resulting possibilities are far more than with flight hardware. New experimental pages can be added very quickly. Graphical formats can be used on the screen of the CDU and the function of the keys is fully programmable.

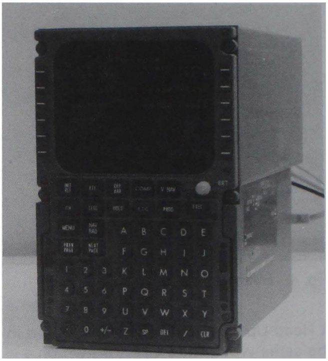


Fig. 7 Simulator CDU as used in RFMS

The first experiment utilizing the flexibility of this FMS simulation, also known as Research Flight Management System (RFMS), was the FAA Data Link experiment, where the CDU was one of the man-machine interfaces under investigation for operating the ATC Data Link[6]. For this experiment a complete ATC menu has been developed and implemented in the FMS software, linking FMS route data to the ATC datalink pages.

The advantage of the in-house written Boeing 747-400 FMS simulation program is obvious: new data link pages are easily added. Also a new functionality, the data link interface, was added to the RFMS program. This year the interface has been expanded with another data link protocol and a series of automatic functions to transfer ATC clearances to the FMC. Later this year a follow-up of the first datalink experiment will take place. The main topic of this investigation will be the human factors of the man-machine interface of the data link. The interface will now be more automated than in the first experiment. The RFMS is now a standard feature in the full glass cockpit of the NLR RFS and is used in every experiment and for different aircraft types.

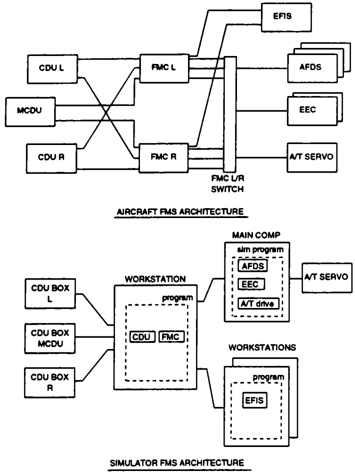


Fig. 8 Difference between aircraft and research simulator in architecture FMS

CONCEPT

In the example above there is one common characteristic feature, which is typical for a good research flight simulator. This is the so-called 'smart software-simple hardware' concept. This means that the complex functions and logics should be dealt with in the software as much as possible, in this way reducing the complexity of the hardware.

For example in case of the FMS, the CDU used in the RFS is nothing more than a display and switches. All the logic including scratchpad functions and the drawing of the characters is performed by the simulation software. The same goes for the other two examples: EFIS/HUD and Flight Mode Panel. In real aircraft, hardware units (incl. software) are more autonomous communicating via the ARINC bus.

The basic idea is that in general software is more flexible than hardware. Changing or expanding a program, recompiling and using it is easier, quicker and cheaper than 'bending metal'. This is especially true, if the software is developed in-house.

CONDITIONS

Next to the simulation vs. stimulation dilemma, there now is the question: how much can be dealt with in the software? For flexibility as much functions as possible should be dealt with in the software. Doing this, attention should be paid to the following matters:

a. Is the program/computer speed sufficient?
b. Are the data available and are they sufficient?
c. Is the know-how to simulate this system with 'smart software' present?
d. Will the final result be realistic?
e. Is the 'simple hardware' available?

ad a. Is the program/computer speed sufficient?

One of the reasons software can now simulate a range of hardware systems is the enormous development of the processing power of today's computers. This enables the computer to perform complex functions in a small time span, enabling a loop frequency, which seems continuous to the user. The speed requirements strongly depend on: how is it sensed by the user, how fast is the real system you want to simulate, how complex is the real system?

ad b. Are the data available and are they sufficient?

To build a software simulation of a hardware system, a lot of data inlcuding knowledge of the logic of the system is needed. Obtaining the data is always difficult. There are different ways in obtaining the required data: from the black box approach to a co-operating manufacturer. Sometimes even the manufacturer is not able to answer all questions when building a software simulation, because he only builds the hardware solution.

ad c. Is the know-how present to build the software simulation?

A major research institute has the specialists needed. Or they have the budget to do research in the area needed. If this is not possible, there are two solutions: choose the (flight) hardware or hire subcontractors to write the software. One should keep in mind that exploiting the main advantage of software, quickly and cheaply changable, could become difficult or impossible in case of software written by subcontractors, even with proper documentation.

ad d. Will the final result be realistic?
The realism of the flight deck in the final solution should be as high as possible. This in case of the NLR RFS was often the main factor in determining the line between software/hardware functions. The advantage of modern, full glass cockpits is that they use the same technology as was already used in simulation. In the past, some simulators used to draw analog instruments on computer displays, while nowadays the same computer displays are used in the cockpit of the aircraft.

ad e. Is the 'simple' hardware available?
Though the functions of the hardware may be 'simple', the hardware becomes highly specialized. If this hardware cannot be purchased, it sometimes has to be developed by the research institute. This again is no problem for a major institute. The main issue here is: how to make it look realistic, while it is only a collection of switches, lights and/or computer displays. (see example of CDU)

If all these conditions are met, the 'smart software-simple hardware' concept results in a very efficient research flight simulator.

CONCLUSION

In the NLR Research Flight Simulator, where the 'smart software-simple hardware' concept has been in use for years now, we see the result: The simulator cockpit has evolved into a full-glass cockpit with all features of a modern aircraft cockpit: EFIS, Multi-Function Displays, FMS and an optional side-stick controller. The transport cockpit is not only up-to-date but even ahead of its time. It includes futuristic elements like automated ATC Data Link interfaces, new fly-by-wire concepts, a head-up display, Take-Off Performance Monitor (TOPM) display[7], Experimental Flight Management System for 4D guidance. All come together with the standard features of motion, two different visual systems, data recording, FMS, EFIS, touchscreens, radio control panels, optional analog displays and physiological measurements. The application of the 'smart software - simple hardware' concept yields a flexible simulator. Increasing the versatility immediately broadens the scope of the research potential of the simulator. And only that determines the value of the flight simulator as a research tool.

REFERENCES

1. NLR Research Flight Simulator Transport Cockpit Reference Guide; J. Kossen, J.M. Hoekstra; NLR TR 93441L; October 1993; NLR

2. Fokker 100 EFIS Software Requirements; Anon.; Rockwell-Collins, Doc. 634-1693-012

3. Testplan Investigation on the NLR Flight Simulator concerning Handling Qualities of Transport Aircraft with Advanced Flight Control and Display Systems; W.P. de Boer, C. La Burthe et al; NLR TR 87060L; April 1987; NLR

4. Testplan F-28 Mk0100 EFIS Evaluation; M.F.C. van Gool; NLR TR 86084C; August 1986; NLR

5. The NADDES DDF format; P.J. Hoogeboom; April 1994; NLR

6. Flight Simulator Evaluation of Base Line Crew Performance with Three Data Link interfaces; R.N.H.W. van Gent, H.G.M. Bohnen et al; May 1994; NLR

7. Flight Simulator Evaluation of Take-Offs conducted with and without a Take-Off Performance Monitor (TOPM); R. Khatwa, J.J.L.H. Verspay; 19th ICAs conference September 1994

AIAA-94-3417-CP

ADVANCED MISSILE CONCEPT EVALUATION TOOLSET (AMCET)

C.M. Ewing
Wright Laboratory Armament Directorate, Eglin AFB FL

Abstract

This paper discusses the Advanced Missile Concept Evaluation Toolset (AMCET) developed at the Wright Laboratory Armament Directorate (WL/MN). This set of analysis tools is used to support detailed ground testing of air-to-air. air-to-surface, ground-launched endoatmospheric, and exoatmospheric missile systems. It includes six-degree-of freedom (6-DOF) digital simulation; synthetic ultraviolet, visible, infrared, LADAR, and millimeter wave image generation; hardware-in-the-loop; (HWIL) and lethality effectiveness analysis. These four assets provide WL/MN a unique capability to examine the performance of advanced guided munition concepts. The AMCET can be used for everything from first order kinematic accessibility studies to hardware-in-the-loop analysis of seeker, signal processor, and guidance hardware/software.

1.0 Introduction

Since the funds allocated to perform flight tests of advanced weapon systems continue to decrease, non-destructive experiments of these systems will become an increasingly more desirable method of evaluation. Over the past ten years, a set of analysis tools has been developed at the Wright Laboratory Armament Directorate which can be used to support detailed evaluation of many types of advanced missile concepts. The Advanced Missile Concept Evaluation Toolset (AMCET) consists of detailed 6-DOF digital simulation, synthetic image generation, HWIL, and lethality effectiveness.

This analysis capability is supported by a rich heritage in guided weapon system development for the Air Force (e.g. Advanced Medium Range Air-to-Air Missile (AMRAAM), the Laser Guided Bomb series (GBU-10, 12, 24, 27, 28), GBU-15, and AGM-130 series). Additionally, WL/MN has supported the Ballistic Missile Defense Organization (BMDO) in the Space Based Kinetic Kill Vehicle (SBKKV), Space-Based Interceptor (SBI), Light-weight Endoatmospheric/ Exoatmospheric Projectile (LEAP), Endo/Exoatmospheric Interceptor (EEI), RAPTOR/TALON, and the Theater High Altitude Area Defense (THAAD) programs. WL/MN has developed many advanced technology components, including seekers, gyros, airframes, and guidance systems, for over 25 years.

The AMCET architecture is shown in Figure 1. Each tool is capable of generating useful analysis by itself as well as being incorporated in the toolset to provide a complete picture of the guided missile's performance from first order analysis to hardware performance.

2.0 Digital 6-DOF Simulation

Historically speaking, digital simulation has always been a valuable means of analyzing weapon concepts in the early stages of development. Digital simulation activities at WL/MN include kinematic accessibility studies, system and sub-system component analysis, and conceptual design trade studies. In addition to analyzing missile systems, digital simulation

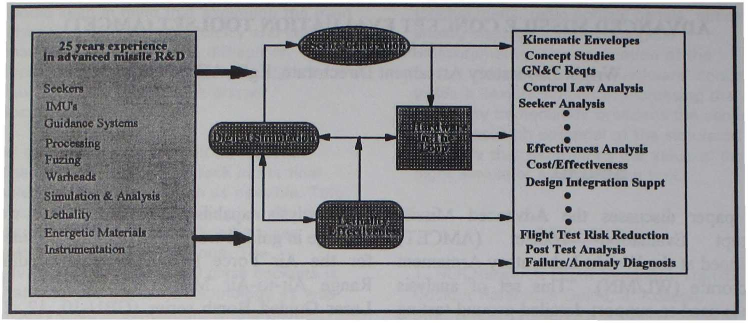


Figure 1 - AMCET Methodology

is also used to generate geometry data and dynamics files for both the missile and target for use in synthetic scene generation. Over the years WL/MN has used many simulations to model various missile concepts. The current simulation in the AMCET used for 3-DOF and 6-DOF analysis is AVATAR, developed by Science Applications International Corporation (SAIC) Orlando. A very flexible, high fidelity simulation, it provides the means for analysis ranging from first order kinematic accessibility to detailed modeling of guidance subsystems in a 6-DOF engagement.

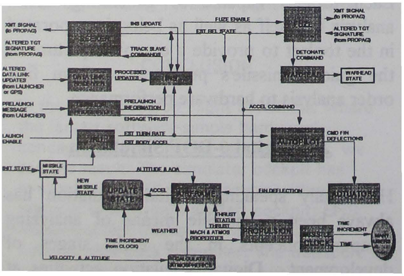


Figure 2 - 6-DOF Block Diagram

AVATAR was constructed to provide maximum flexibility in simulating the dynamics of missile-target engagements with user controlled inputs. It allows for rapid modeling of missile concepts due to the modularity of its components. Standard components are stored in a missile library from which they can be used directly or modified to more accurately model a particular vehicle. Figure 2 illustrates some of the basic components of missile and target simulation (propulsion, aerodynamics, mass properties, sensors, guidance and control, autopilot, actuator, etc.) which have been packaged around unique program sequence control methods. AVATAR is currently being used on a VAX 4000 system. There are also a number of other detailed simulations developed to support specific weapon systems in use at WL/MN.

Currently WL/MN is heavily involved in bringing our digital simulation environment in line with the Joint Modeling and Simulation System (JMASS). JMASS is a joint Air Force, Army, Navy, and ARPA program to provide a standard modeling and simulation environment to foster software reuse and language independence. It combines analytic/ constructive (conceptual), dynamic/interactive (man-in-the-loop), and emulative/high fidelity simulations into a complete analysis package. It adheres to government recognized programming and networking standards and will support all phases of the defense system acquisition cycle. Figure 3 shows the JMASS system.

WL/MN has been selected as a beta site to test the JMASS architecture for weapon system modeling applications. To accomplish this the Modular Munition Simulation (MOMS) architecture and taxonomy are being developed to be used within the JMASS system. It will be a reusable, extendible set of missile subsystem model specifications in Ada, allowing diverse simulations to be built using compatible subsystem parts.

Besides providing a standard software architecture, JMASS also offers a sophisticated simulation development tool. In a windows environment, components can be selected, initialized and configured as desired. Simulation can be done in interactive or batch mode and the data output configuration will be flexible and comprehensive. A data visualization tool displays simulation runs in animated style using 3-D graphics. Other analysis tools provide plotting, data conditioning, transformations, comparisons, and anomaly investigations. Figure 3 shows the JMASS simulation environment being used in a windows environment. WL/MN expects to have both air-to-air and air-to-ground simulations operational in the JMASS environment by September 94. The MOMS architecture runs in the JMASS environment on either a Sun Workstation or the IBM RS 6000. It is currently being ported to other platforms.

Using the digital simulation environment described above, and the expertise in component development and modeling which exists at WL/MN; analysis of many missile concepts has been completed with high confidence in the results. Some of recent missile concepts investigated include AMRAAM, PHOENIX, LOCAAS, RAPTOR/TALON, Peregrine, Hyper Velocity Missile, LEAP, and THAAD. In addition to digital simulation of their flight characteristics, much work has also gone into analyzing sensor performance for these and other advanced homing missiles.

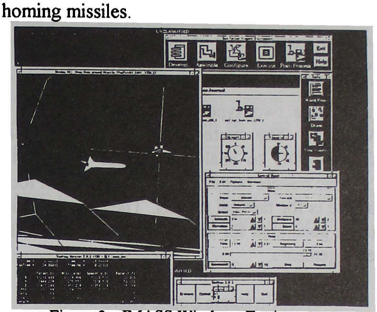


Figure 3 - JMASS Windows Environment

3.0 Synthetic Scene Generation

As the sensors on missiles have become more and more complex, the need for an accurate means of testing them on the ground has also grown. To meet this need, WL/MN has developed a world class synthetic target signature generation capability. Using the missile and target dynamics generated with the digital 6-DOF simulation, target imagery is generated against various backgrounds for both tactical and strategic targets. The imagery, in the form of single snapshots or movie sequences, has proven useful by providing realistic data to support design, performance prediction, validation, and trade studies for components and complete systems. This imagery is also available for use in HWIL simulation as the actual seeker hardware is exposed to the synthetic imagery by way of in-band scene projectors or direct digital injection.

In computing target signatures, many physical models are used to represent the signature contributions during an engagement. For commonly used infrared (IR) seekers, these contributions could include thermal emissions from solid bodies, in-band emissions from target exhaust plumes, emissions from flow fields and wakes, and scattering of external source energy from the body and particulates associated with plumes or ablation products.

There are two types of scene generation at WL/MN. The strategic tools were developed for BMDO to analyze strategic and theater ballistic missile targets. The second set of tools in use is for tactical targets. They allow scenes to be generated for air-to-surface and air-to-air engagements.

3.1 Strategic Scene Generation

Part of the scene generation capability comes from government-sponsored development of several community standard models which implement detailed mathematical descriptions of important signature phenomena.

For National Missile Defense scenarios, WL/MN employs the Strategic Scene Generation Model (SSGM). SSGM is a compilation of industry-standard IR phenomenology models which are integrated to produce synthetic IR imagery of ballistic missile booster bodies, rocket engine plumes, and associated objects (such as decoys, balloons, and re-entry vehicles). To include background effects, SSGM also includes models of terrain, clouds, atmosphere, aurora, space, and nuclear effects.

The SSGM models do not include some important features that exist in Theater Missile Defense (TMD) scenarios or in situations with the interceptor/observer positioned within the discernible atmosphere. For these conditions, WL/MN has been integrating TMD oriented models into the Eglin Scenario and Signature Program (ESSP). Figure 4 illustrates the programs and the flow contained in ESSP. Notice that elements of the integration are appropriate to both the threat and the interceptor, since both typically fly hypervelocity trajectories within the atmosphere. Cornerstone components of ESSP include the Aerothermal Analysis Program (ATAP) and the Composite High Altitude Maneuvering Program (CHAMP). ATAP is an in-house hardbody thermal response code which calculates temperatures on a missile body given the body geometry, material composition, and trajectory. CHAMP is a leading edge model in IR scene generation. It is a radiance model and image pixalizer which employs hardbody temperatures, flowfield radiance, and external sources (such as solar flux, or plume impingement) to create synthetic images or scene sequences. The CAD-like features of CHAMP geometry models allow the user to specify complex geometries, such as finned missiles or post-boost vehicle configurations.

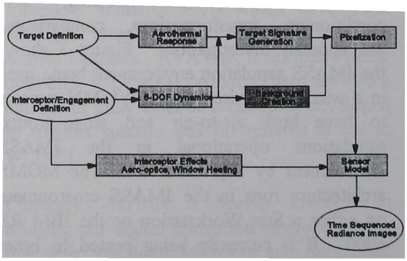

Target Definition
Aerothermal Response
Target Signature Generation
Pixelization
Interceptor/Engagement Definition
6-DOF Dynamics
Background Creation
Interceptor Effects Aero-optics, Window Heating
Sensor Model
Time Sequenced Radiance Images

Figure 4 - ESSP Block Diagram

In addition to the ability to fully model the target phenomenology, WL/MN has recently been developing the capability to incorporate sensor window effects into the scene phenomenology. With some current missile concepts flying at hypervelocity speeds, they are burdened with thermal protection to shield the sensor, electronics, and airframe from the aerothermal environment. These thermal protection devices can consist of ablative noses, active coolant flows, and sensor window heat resistive materials. The effects associated with the hypervelocity flowfield and its interaction with the airframe can cause noise and distortions in the perceived signature as the seeker looks through them.

Work is ongoing at WL/MN to model these critical error sources and incorporate the effects into the scenes to account for blurring, boresight error, flowfield and window thermal noise. Currently all of the scene generation work is being done on Sun and Silicon Graphics Workstations.

3.2 Tactical Scene Generation

In addition to the strategic scene generation capability, an equally in-depth ability to generate tactical scene imagery also exists at WL/MN. Due to space limitations, this capability will only be briefly described. WL/MN has developed the capability to generate IR, millimeter wave (MMW), and RF signatures for air-to-air and air-to-surface engagements using several varieties of tools.

The Data Analysis and Modeling program (Irma), developed by WL/MN was one of the first signature prediction models created for the generation of high resolution synthetic IR target and background signatures for tactical weapon scenarios. It is used to generate IR scenes for smart weapons research and development. It is being upgraded to include the ability to model active and passive IR, ladar, and active and passive MMW for air-to-surface engagements.

RF signatures for air-to-air engagements are modeled using the RF Signature Program (RFSig). Using triangular faceted target models, phenomenology in the far-field X-band can be simulated. It is currently being upgraded to include the W-band.

The Spectral IR Imaging of Targets and Scenery (SPIRITS) code is in use to generate passive IR imagery for air-to-air engagements.

Once the scene generation models have been used to generate the individual scenes the Synthetic Scene Workstation (SSW), developed at WL/MN, can be used to format the data as needed. Scenes can be used to aid in signal processing algorithm development, seeker design, or be fed through an in-band scene projector for HWIL testing of seekers.

4.0 Hardware-In-The-Loop

The Kinetic Kill Vehicle Hardware-in-the-Loop Simulator (KHILS) facility is an ongoing development by WL/MN to provide an independent national resource for the accomplishment of non-destructive HWIL performance testing of missile systems and subsystems. The KHILS facility was originally built for BMDO strategic interceptor analysis, however it is quite applicable for tactical weapon system testing as well. It is specifically designed to concentrate performance analysis efforts on seeker, signal processing, and GNC subsystems over a launch to impact flight scenario.

The overall function of KHILS is to simulate one-on-one engagements using missile hardware such as the seeker, signal processor, and guidance processor. The simulation exercises the flight hardware and software in all engagement flyout phases, including pre-launch, initialization, pointing, launch, midcourse, acquisition, and terminal homing to

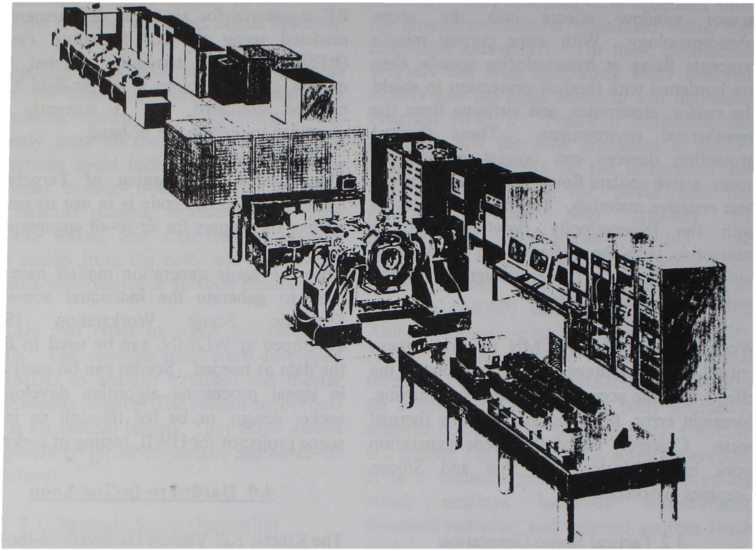


Figure 5 - KHILS Facility

impact. It can also be used for open loop testing of stand-alone missile subsystem components. Figure 5 shows the facility layout while Figure 6 is a block diagram of how the facility functions. As illustrated in Figure 6, a real time closed loop simulation includes actual missile hardware and software integrated into the KHILS test environment to perform the majority of the ongoing GNC functions. The KHILS Physical Effects Simulators (PES) provides the external stimuli to excite the seeker with images that match, as closely as possible, an actual engagement. The KHILS PESs include: the scene projection system (to present in-band imagery to the seeker), the image generation system (to provide digital representations of targets and backgrounds to the projectors or for direct injection to the missile signal processor), and the Flight Motion Simulator (FMS). All of these simulators must function in a real-time, closed-loop environment with minimal contribution to transport delay of signals into and out of the hardware. Due to space limitations, each component of the PES will only be briefly described.

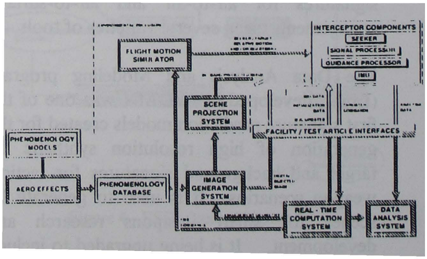


Figure 6 - KHILS Block Diagram

4.1 Scene Projection Systems

The scene projection system consists of four high-fidelity scene projection systems. These include the Scanning Laser IR Scene Projector (SL-IRSP), the Visible Spectrum Scene Projector, and the Resistor Array Prototype (RAPP).

WL/MN has led the nation in sponsoring research in projectors. Several prototype devices are in use and a current development program, the Wide-Band IR Scene Projector (WISP), will build a device with extremely broad application to include evaluation of tactical systems.

4.1.1 Scanning Laser IRSP

The SL-IRSP is designed to simultaneously project high resolution imagery in three IR wavebands with a frame rate in all bands of 400 Hz. The IRSP uses laser energy and a Scophony scanning modulation approach to meet this requirement. The system contains five separate optical trains. These trains can be seen in the IRSP optical layout shown in Figure 7.

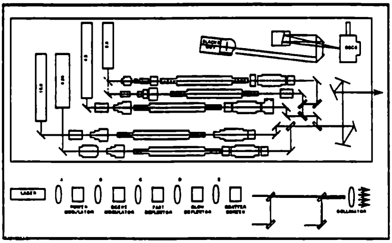


Figure 7 - IRSP Optical Layout

4.1.2 Visible Target Simulator

The Visible Target Simulator (VTS) is shown in Figure 8. The video information of the target scenes is sent to a CRT, but instead of using the visible source to drive a Liquid Crystal Light Valve (LCLV) that turns the visible image into IR, the LCLV and blackbody source are removed and the dynamic target scene is transmitted directly to the seeker in the visible waveband.

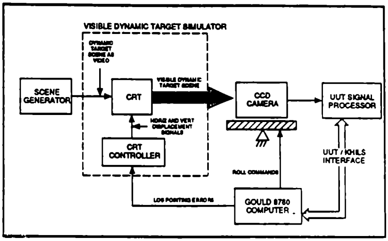


Figure 8 - Visible Scene Projector

4.1.3 Resistive Array Prototype

The RAPP simulator, seen in Figure 9, provides KHILS with broadband, flicker-free, realtime, dynamic IR scene projection capability for testing imaging systems operating in the 3-12 micron waveband. The system consists of four major components; the interface and control electronics which translate 12 bit digital scene data into analog electronic signals; the resistor array and mount which converts analog electronic signals into IR images; the collimator (two sets) which projects the IR scene images onto the seeker under test; and a Sun workstation that serves as the user's console. The flickerless quality of the images greatly simplifies the synchronization between the RAPP and seeker under test. The system electronics limit the RAPP to 400 frames per second although the resistor array temporal response suggests a frame rate of 1000 frames per second or more.

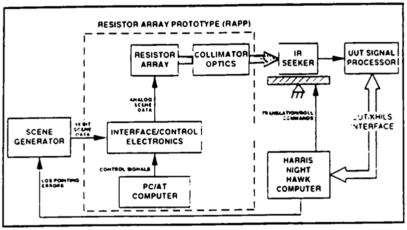


Figure 9 - RAPP Functional Block Diagram

4.2 Scene Generation RAM

The Scene Generation Random Access Memory (SGRAM) system was developed to provide real-time scene playback for the KHILS facility. The two-dimensional, prestored scenes can represent IR, UV, or visible imagery.

The real-time NightHawk computer provides target LOS data to SGRAM at the simulation frame rate. These data include target X, Y translation, target rotation angle, and a parameter to indicate target range. SGRAM uses the range term to look up the proper frame of digital imagery from memory. This selected frame of data is then modified according to the commanded translational and rotational parameters and output to the test article.

SGRAM can then be used to provide digital imagery to one of the scene projectors described above or can provide direct digital signal injection to a signal processor, thus bypassing the sensor optics, focal plane array, and readout electronics.

4.3 Flight Motion Simulator

The flight motion simulator is a Carco Electronics model S-457 three axis device, developed specifically for the KHILS facility. Table 1 illustrates the performance specifications for the FMS. Important design features include a 60 Hz bandwidth in each of the three axes and sub-pixel pointing accuracy sufficient to support closed loop HWIL simulation of high resolution strapdown seekers.

	ROLL AXIS	YAW AXIS	PITCH AXIS
SPECIFICATION			
LOAD SIZE - 15 cm dia. x 60 cm long			
LOAD WEIGHT - 12 kg			
LOAD INERTIA, KG - m	0.03	0.5	0.5
MAXIMUM ACCELERATION, °/sec	35,000	18,000	18,000
MAXIMUM VELOCITY, °/sec	1,000	1,000	1,000
POSITION COMMAND, ANALOG			
DISPLACEMENT, degrees	±120	±10	±10
MINIMUM VELOCITY, °/sec	0.0004	0.0004	0.0004
FREQUENCY RESPONSE, 0.25° p - p command			
- Phase Shift at 20 Hz, degrees	<5	<5	<5
- Phase Shift at 60 Hz, degrees	<90	<90	<90
REPEATIBILITY, degrees, Maximum	0.005	0.005	0.005
DRIFT, Maximum (one hour), degrees	0.005	0.005	0.005
THRESHOLD, degrees	0.0003	0.0003	0.0003
POSITION ACCURACY, degrees	±0.05	±0.05	±0.05
POSITION RESOLUTION, degrees	0.0003	0.0003	0.0003
READOUT ACCURACY, Maximum, degrees	±0.05	±0.05	±0.05
VELOCITY COMMAND, ANALOG			
MINIMUM VELOCITY, °/sec .25°	0.004	0.004	0.004
FREQUENCY RESPONSE, [illegible]/sec p - p command	60	60	60
ACCURACY %	1	1	1
VELOCITY RIPPLE, Maximum, %	0.75	0.75	0.75

Table - 1 FMS Performance Limits

4.4 Computational Resources

The major facility computational resources consist of the Real-Time Computer System (NightHawk 5800), the Data Analysis and Visualization System (SGI Onyx Reality Engine II), and the real time data display system (SGI VGX), along with various support equipment for graphing, plotting, and display of results.

The HWIL facility provides an independent national test resource for the accomplishment of non-destructive HWIL testing of precision guided weapon systems and subsystems. It provides an important piece of the complete simulation and analysis capability developed at WL/MN, as it provides the basis for developing high fidelity digital simulation models of components and guided munition systems.

5.0 Lethality Assessment

Running simulations on various missile concepts only verifies that the missile can reach its intended target. Once there it must be able to kill it. Over the last 10 years WL/MN has developed a premiere lethality assessment capability for strategic targets which combines with more than 25 years of experience in tactical lethality work. A database consisting of over 400 hypervelocity impact experiments which utilized component, sub-scale, and full-scale target models has been compiled.

In addition to performing lethality experiments, WL/MN has a combination of tools for use in simulating target lethality. These include the Effectiveness and Vulnerability Assessment (EVA) program (designed for ground fixed targets), the Point Burst Damage Assessment Model (PDAM) program (designed for ground mobile targets), and the Kinetic Energy Analytic Projectile Effects Program II (KAPPII) (designed for missile targets). The development and validation of these models is done through scale and sub-scale lethality experiments conducted at WL/MN.

EVA-3D assesses the survivability/vulnerability of hardened (fixed ground) targets and the interaction of conventional weapons with those targets. It does this by modeling the attack or delivery conditions, the weapon trajectory, fuzing, and burst damage inflicted on the target in a Monte Carlo approach. Targets capable of being analyzed include buried, hardened structures such as communications, command, and control facilities as well as aircraft shelters.

The PDAM code was designed to predict damage to armored (ground mobile) vehicles caused by antimateriel weapons. It does this by determining damage at a component level and using fault tree methodology to combine the component damage values into an overall prediction of damage to the target. High density kinetic energy penetrator, shaped charge, and high explosive warheads can be modeled with this code. Multiple failure modes for each component can be evaluated using the most meaningful predictor of failure. Using a functional failure mode fault tree to aggregate component functional failure, PDAM predicts the probability of achieving mobility, firepower, or catastrophic levels of damage for each weapon impact.

KAPII is a set of fast-running algorithms developed to predict damage to complex 3-D aerospace (missile) targets impacted by single or multiple kinetic energy weapon projectiles that are represented by chunky fragments, rods, and hollow rods. The code is a fusion of previous similar codes developed under the Defense Nuclear Agency's Lethality and Target Hardening (LTH-5) program for both the Air Force and Army. KAPII is validated with an extensive experimental database covering a range of impact conditions with velocities from 1 to 8 km/sec. The algorithms describe the target's reaction to the projectile effects and include penetration and hole formation, high explosive initiation, component dismemberment, hydraulic ram effects, and other damage modes. It also characterizes the state of the projectile as it progresses through the target and includes residual mass, residual velocity, and fragmentation.

The next and most critical step in the process of determining kill effectiveness is to perform a fault tree analysis. The fault tree analysis is a study of the critical electrical, mechanical, hydraulic, and black box systems in the target. WL/MN has developed the expertise to understand how the target components operate and how the damage to each target will effect the probability of kill for the entire target.

By incorporating the fault tree analysis and the damage potential of a weapon system into the appropriate engineering model, a high confidence lethality assessment for a particular weapon/target pair can be performed.

The above analysis codes are used on a computational suite consisting of a SGI Challenge R4400, a SGI Iris graphics workstation, and a SGI Indigo R4400 workstation.

Concluding Remarks

With increased reliance on simulation and analysis to determine which future weapon systems will be developed, WL/MN has risen to the challenge of providing a comprehensive capability to provide independent, detailed, and accurate information to meet the customers' need for analysis information. Through the use of digital simulation, synthetic scene generation, HWIL, and lethality assessment, accurate judgments can be made to determine the feasibility of a proposed weapon concept.

WL/MN will continue to build its weapon assessment capability over the years to meet the changing needs of the modeling and simulation community.

Acknowledgments

I would like to thank the Seeker Technology Evaluation Branch (MNGI), the Technology Assessment Branch (MNSA), and the Simulation and Assessment Branch (MNSH) for their cooperation in putting this paper together.

The Influence of Platform Mass Properties on Simulator Motion System Performance

S.K. Advani [1], R.J. Verbeek [2]
SIMONA International Research Centre
Delft University of Technology
The Netherlands

Abstract

The dynamic performance of a simulator motion system is to a great extent determined by the mechanical qualities of the motion system hydraulics, the motion control system which provides signals to drive the actuator servo valves, and also the mass properties of the moving equipment of the simulator placed upon the motion system. Quite commonly, in the case of training flight simulators, the centre of gravity of the large moving payload is located far above the motion system's upper gimbals. An investigation, using a complete dynamic model of the hydraulic motion system, was carried out to quantify the effects of payload mass properties on motion system performance. This paper will review the results obtained from these studies. It is found that the total mass and the vertical location of the centre of gravity influence the time delays, phase and parasitic noises of the motion system, and that these properties also limit any further improvements due to design changes in software or hydraulic hardware. By adapting these design considerations, a motion system with inherently higher performance will result. Therefore, a fundamentally new and yet practical light-weight motion platform design is proposed. This configuration is being introduced in the multi-functional SIMONA Research Simulator of the Delft University of Technology.

Introduction

The essential purpose of piloted flight simulators is to reproduce the physical and environmental cues as closely as possible to those encountered in flight. These cues result from control inputs exercised by the pilot, and are also due to external disturbances acting upon the vehicle. The pilot-vehicle interaction forms a closed loop in which the pilot receives information about the state, and change in state of the vehicle, through the sensory organs. The visual information is obtained through the foveal and central vision, while motion cues are derived from the human vestibular system. These stimuli are processed by the brain, which can be thought of as to compare the signals to an internally-developed model. In order for the response of two "similar" systems to match (hence, implying the fidelity of the simulation), the input signals must match to a level below a threshold of perception[1].

In human control tasks, visual information requires longer processing time than the vestibular cues. In other words, the onset of motion is primarily detected by the vestibular system, while long-term changes in motion are sensed predominately by the eyes[2]. This requires that the initial response of the simulator motion system match that of the vehicle being simulated. This "onset cue" often includes the high-frequency components. In an ideal situation, the latency of the simulator should be zero, over the entire frequency range of the vehicle motion. This applies for the presentation of all cues, particularly motion and visual. In practice, this is not easily achieved, mainly due to the limitations in the mechanical hardware and motion control software which induce the sensation of motions to the pilots.

[1] Assistant Professor, SIMONA Project Leader Member AIAA

[2] Research Assistant

servo valve output signal i_v. Within the controller, three types of feedback signals are applied: Position feedback provides stability. The hydraulic actuator otherwise acts as an integrator, converting the servo valve input signal to a mechanical position. Actuator force provides damping, since hydrostatic actuators in particular are poorly damped. Acceleration and/or velocity feedback provide further damping, if desired.

Validation of Simulation Model

The simulation model of the six-degrees-of-freedom motion system was validated in two stages. Initially, standard input signals were given to the system and the outcome checked with expected results. Secondly, a proof-of match for a single actuator was conducted[5]: time histories of the response of the actuator model were thereby compared to the responses of a commercially-available motion system actuator. With a reasonable degree of confidence in the simulation model, the sensitivity study of inertial properties could be conducted. This research and the results obtained will now be discussed.

Inputs to the model

Motions of the platform in single degree of freedom were selected for this evaluation. The synergistic motion system selected represented the kinematical properties of the SIMONA Research Simulator, currently being built at the Delft University of Technology[3]. Its kinematics are typical for current six-degree-of-freedom simulators, requiring the simultaneous movement of all actuators in order to move the platform in any unique degree of freedom. The platform was modelled as a solid sphere with a diameter of 4.0 metres, and having a uniform mass distribution.

One of the most demanding requirements for the simulator motion system is the acceleration in the surge (forward) direction. Moreover, the surge motion imposes the highest dynamic load upon the actuators, particularly towards the end of their stroke. It has been shown, for some configurations, that the dynamic load component on each actuator acting along its centre line can exceed the total weight of the motion platform[8]. This also underscores the need for accurate control over the motion system throughout the entire envelope.

The surge direction was also selected so that the effect of coupling in the pitch degree-of-freedom of the vertically-displaced centre of gravity could be examined.

The parameters which were given as inputs to the model were as follows:

- input acceleration doublet in surge (X) of 1.0 m/s^2, with a cycle of 1.0 seconds.

- variations in mass: 1000 kg, 2000 kg, 4000 kg, and 6000 kg

- variations in vertical location of centre-of-gravity (above upper gimbal plane): 0.0 m, 0.5 m, 0.75 m, 1.0 m, 1.5 m

In order to avoid instabilities to the motion system, it was necessary to vary the feedback gain of the controller, in accordance with the stabilization routine in Reference 7. This suggests that it is indeed difficult to isolate completely the effects of platform mass and inertia, however future research will address this problem in greater detail.

Simulation results

The time histories of the simulator platform were generated to show its response to acceleration to a step-input surge acceleration signal of 0.5 m/s^2. The results of these inputs are now presented.

Variations in total moving mass

Figure 5 shows the responses of the platform due to variations in the total mass of the moving load. Note that the centre of gravity of the moving platform is fixed in the plane of the upper gimbals. This figure clearly indicates that increasing the mass leads to a rapid increase in the latency of the initial response, as the platform natural frequency decreases. Damping, however, increases proportionally with the mass.

Later, the relationship between the mass and the damping and natural frequency will be discussed.

The increased damping may be classified as one of the few advantages of a system with a large mass, however any reduced damping can easily be compensated in the control algorithm, while the undesirable time delay in the onset of the acceleration is tedious to resolve in the control algorithms.

The motion system natural frequency in surge decreases exponentially with increasing mass. At 6,000 kg, the natural frequency determined by this simulation was 19.5 radians per second, clearly within the frequency spectrum of aircraft simulation. A higher natural frequency would suggests a more responsive platform, with a higher system stiffness.

Furthermore, increases in the gross moving load had a significant influence on the stability of the system, and large variations had to be made to the control feedback gains in order to achieve stability. This is quite normal, and is required during the tuning of the motion system controller.

Variations in centre of gravity location

Responses of the simulator platform to variations in the vertical location of the mass centre are shown in Figure 6. The onset of the motion is not significantly delayed by a higher centre of gravity. The centre of gravity location does, however, cause a simultaneous decrease in **both** the damping and the natural frequency. This rate of decrease appears, from this study, to be reasonably linear and can be attributed to the virtual mass (and not the real mass), caused by the dynamic load on the actuators. Unlike a true increase in the gross moving mass, the increase in the vertical offset in the centre of gravity increases a moment coupling effect, hence increasing (unnecessarily) the load on the actuators.

Therefore, a motion platform with a lower centre of gravity appears to have significant advantages: While decreased damping can be compensated, the higher hydraulic natural frequency lends itself to a more responsive system.

Parasitic noise in pitch degree rotation

The parasitic noise in the pitch angular acceleration (q_{dot}) as a result of the input signal in surge is shown in Figure 7 for variations in mass (with the centre of gravity fixed in the platform gimbal plane), while the same for variations in the vertical location of the mass centre are shown in Figure 8.

These responses show that a lighter platform is favourably less sensitive to the parasitic coupling effects in the pitch degree of freedom. As the mass is increased, the initial overshoot also increases (unlike the overshoot in the desired degree of freedom). The damping increases with greater mass, while the natural frequency decreases.

In the case of variations in the vertical centre-of-gravity location, again, the lighter platform yields lower parasitic motions in pitch, while the natural frequency is higher.

Ideally, there should be no pitch coupling, and any minimal effects should be compensated. In the worst case, these parasitic effects should be below their perception thresholds. These arguments support the proposal for light, low c.g. motion platforms.

Application of results

In contributing to the development of a simulator motion system with extremely high inherent performance, the results of this investigation have proven that the payload mass properties indeed require careful attention. The influence of payload mass properties on the dynamic response has therefore provided a guideline for the development of a light-weight structure in which the load-carrying platform and flight-deck are fully integrated. Figure 9 shows the sensitivity of damping and the natural frequency to mass variations, indicating that the desired higher natural frequency is acheived for a lighter platform construction. Furthermore, the results of this study suggest that the mass centre of this structure should be placed as close as possible to the plane of the gimbal attachment points of the motion system actuators to reduce parasitic noise (in pitch) and to attain a high natural frequency.

The realization of the aforementioned requirements is a technical challenge, and represents a change in thought in the design of simulators. Traditionally, the upper platform is the primary structure which carries dynamic loads to the motion system gimbals. The upper platform also carries the cockpit, instructor station and observers chamber, visual display system hardware, and other simulator hardware. Due to the desire to provide commonality, standard platforms are offered by manufacturers, upon which numerous cockpits specific to an airplane type can be built. Furthermore, a client may choose between a number of visual display systems, requiring the platform to be modular so that a wide range of applications can be accommodated without re-design. As a result, a vertically-layered structure is produced. A typical commercial simulator installation is shown in Figure 10.

At the Delft University of Technology, a multi-functional research simulator is currently under development. This device, called the SIMONA Research Simulator (SRS), will serve as a facility for improving simulation techniques for flight and road vehicles, and investigations into man-machine issues of future transportation vehicles. As a simulator tool for fundamental research, the SRS does not require the exact representation of a particular flight-deck or vehicle control station. This provides some flexibility in the specification of the interior volume. Moreover, the SRS is designed as a one-of-a-kind system, thus requiring only one visual display system and few hardware alternatives. On-board hardware is minimized through the careful selection of this equipment. The contribution of electric and electronic cable mass is reduced by incorporating a server system based on a real-time Ethernet protocol, allowing most communications with the host computer to be carried through a single co-axial cable.

The SRS cockpit requires that, in addition to transport airplanes, research into helicopter, automobiles, and other surface vehicles may be conducted. These requirements dictate the form of the interior from an ergonomics standpoint, and the location of the visual display system with respect to the design (pilot-averaged) eye point. In fulfilling these diverse goals, and allowing very rapid changes to the configuration, it is possible to choose from large independent modular units (sub-cockpits), or modular hardware such as instrument panels and controls. Due to the need for a stiff, light-weight construction, the latter was chosen for the SRS.

Lowering the centre of gravity

In order to place the centre of gravity as low as possible with respect to the motion system gimbals while fulfilling the above requirements, the volumes of the control stations for all of the required vehicle types were specified. The geometries of the mechanical equipment, particularly the rudder pedal control loading actuators, were added to the pilot volumes. In defining the bounds of the limits, the maximum kinematic motion envelope of the six actuators was defined through the actuator axis transformation relationships. Note that this envelope included the physical volume of the cylinders. These constraints were used to define the shape of a thin-walled composite structure which would contain the research stations, and also carry the structural loads. This would thereby serve as an integrated platform and cockpit in one construction element. Figure 11 shows the flight-deck in relation to the motion system envelope and the visual display system.

Reduction of Mass

The optimization of the total gross moving load is accomplished in the case of the SRS by constructing the flight-deck shell from advanced composite materials, namely high-temperature curing carbon-fibre and aramids. The careful design and selection of all on-board equipment further reduces the total mass. The experiment control station, or instructor station in commercial simulators, and observer chamber is simplified to only one chair.

A light-weight platform, on the other hand, allows the motion system hydraulic actuators to excite the structure with higher input frequencies due to the increase in available hydro-mechanical power. This requires the platform, flight-deck and all on-board structures to be of high stiffness. By meeting these stiffness requirements, the light-weight platform provides a simulator with an inherently high bandwidth.

Visual Display System

A wide-angle projection-based collimating display system was chosen, allowing the design eye point to be liberally varied within a viewing volume with negligible distortion to the presented image. Monitor-based systems provide a very limited viewing volume, and would limit the flexibility of the pilot position in the interior. In the case of the SRS, it became clear that the display system structure would also have to minimize its contribution to the mass and inertia. Therefore, in cooperation with SEOS Displays Ltd., a new structure for the visual display system incorporating advanced composite materials optimized through Finite Element Analyses, is under development.

Degree of Improvement

The highly-integrated composite materials flight-deck of the SRS offers a generic and yet complete full-flight simulator with a fraction of the weight of most commercial six-degrees-of-freedom training simulators. With all equipment on board, the total weight reaches 2,000 kilograms, while a commercial system (such as for a transport airplane full-flight simulator) may exceed 12,000 kilograms. The centre of gravity, mainly dictated by the height of the visual display system, is 0.6 metres above the upper gimbal plane. Note that this may exceed 1.5 metres in training systems, mainly due to the large, modular components built upon the platform.

The implementation of the findings from this study have been used to develop the SRS hardware. It can be stated that the potential performance of this system, due to the flight-deck and motion system hardware, as well as the controlling software, will exceed the capabilities of current simulator systems. With the proposed design, and with advanced multi-variable, nonlinear and predictive control strategies[3], it will be possible to achieve a virtual zero latency over the entire frequency range relevant to piloted simulations. The SIMONA Research Simulator is shown in an artist's impression in Figure 12.

Conclusions

From the simulation studies conducted in this research, it can be concluded that the mass and the location of the centre of gravity of the simulator gross moving payload both have a profound effect on the behaviour of the system. In particular, the system latency is lowered and the natural frequency increased by reductions in the mass. The natural frequency is higher when the centre of gravity is lowered.

The reduction in weight or the lowering of the mass centre offer significant improvements to simulator motion systems. A system with good natural properties will require less compensation to acheive a desired standard of performance and, with control feedback, will result in a system with superior performance.

While damping is a second order effect (which is possible to fully compensate with a proper control system design), the initial response or system latency is critical to the simulator bandwidth. The initial response is furthermore highly non-linear and cannot easily be predicted. Therefore, in order to reduce the motion system latency in the surge degree of freedom, the platform mass **must** be minimized.

A lower vertical centre of gravity location increases the system natural frequency. Figure 9 showed the effects of mass on the natural frequency and damping. Note that the lower the mass, the more critical the changes in these values.

An increase in either the mass, or its vertical location increases also the amount of undesirable pitch parasitic noise.

It is important for the simulator motion system designer to recognize the negative effects which may result from non-optimized payloads. This research should indicate the merits of optimizing the inertial properties. A higher bandwidth can be acheived without feedback. Eventually, these may contribute to reductions in the acquisition and the life-cycle costs of the simulator due to lighter platform constructions, simpler building foundations and lower energy requirements.

Recommendations

In this preliminary investigation, only the response in the surge degree-of-freedom was evaluated. It would be valuable to quantify mass property effects on other degrees of freedom. The frequency response of the system as a function of the inertial properties would prove most useful in further control law design.

A proper system identification procedure should be applied to the simulation studies, including (as input parameters) any variations to the control system feedback gains. This would provide an accurate measure for simulator motion system and platform design.

References

1. Hosman, R.J.A.W. and Steen, H. van der, "False Cue Detection Thresholds in Flight Simulation". AIAA-93-35 -CP. From AIAA Flight Simulation Technologies Conference, Monterey, August, 1993.

2. Vaart, J.C. van der, "Modelling of Perception and Action in Compensatory Manual Control Tasks". Ph.D. thesis, Delft University of Technology, December 1992.

3. Advani, S.K., "The Development of SIMONA: A Simulator Facility for Advanced Research into Simulation Techniques, Motion System Control and Navigation Systems Technologies". AIAA-93-3574-CP. From AIAA Flight Simulation Technologies Conference, Monterey, August, 1993.

4. Advani, S.K., et al, "What Optical Cues do Pilots Use to Initiate the Landing Flare? Results from a Simulator Experiment". AIAA-93-3568-CP, From AIAA Flight Simulation Technologies Conference, Monterey, August, 1993.

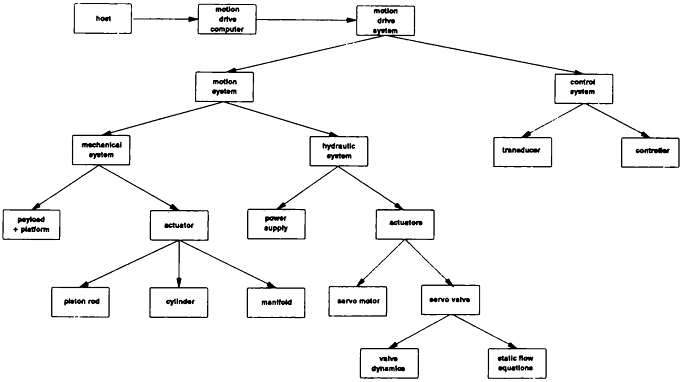
5. Kuit, J.J., "A Nonlinear Simulation Model of a Six-Degrees-of-Freedom Motion System". Delft University Thesis Report, July 1991.

6. Viersma, T.J., Baarspul, M., "Development and Application of a Moving-base Visual Flight Simulator, Including the Design of Hydraulic Actuators with Hydrostatic Bearings". ICAS-80-9.2, 12th Congress of the International Council of the Aeronautical Sciences, Munich, 1980.

7. Viersma, T.J., "Analysis, Synthesis and Design of Hydraulic Servosystems and Pipelines". Lecture Notes 129, Delft University of Technology, Delft, The Netherlands, December 1989.

8. Levi, R.W., and Hayashigawa, L., "Specification Considerations for a Small Motion-Base". From AIAA Flight Simulation Technologies Conference, Atlanta, 1988.

Figure 1. Simulator motion system math model components[5]



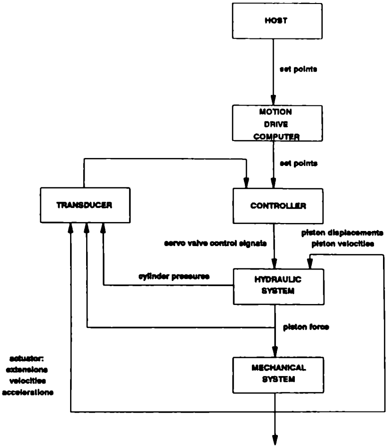


Figure 2. Model structure[5]

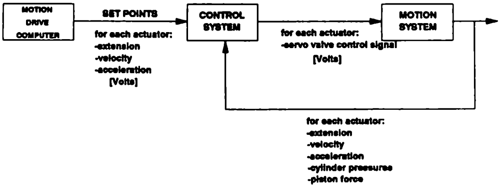


Figure 3. Motion drive system coupling with computer[5]

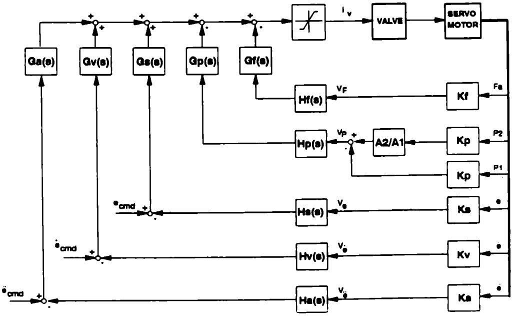


Figure 4. Actuator internal controller[5]

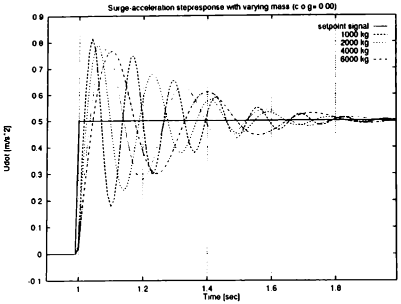


Figure 5.. Effect of mass variations on surge acceleration response

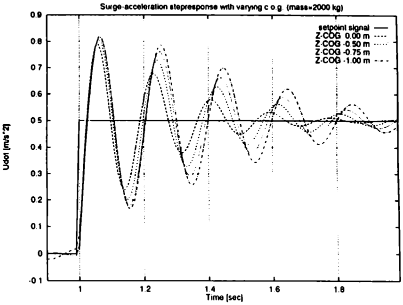


Figure 6. Effect of centre of gravity variations on surge acceleration response

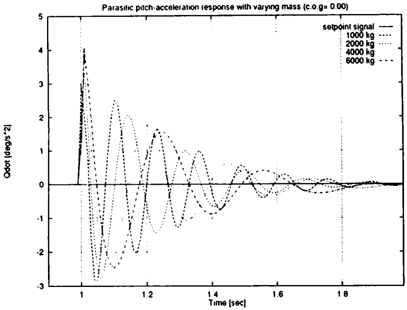


Figure 7. Effect of mass variations on parasitic pitch rotation due to input in surge

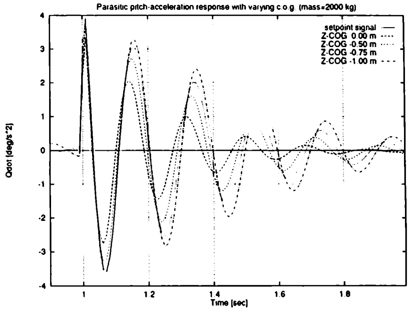


Figure 8. Effect of centre-of-gravity location on parasitic pitch rotation due to input in surge

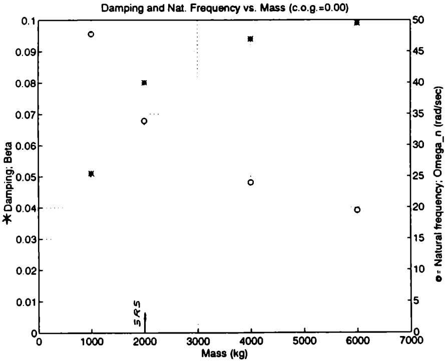


Figure 9. Effect of platform mass variations on damping and natural frequency

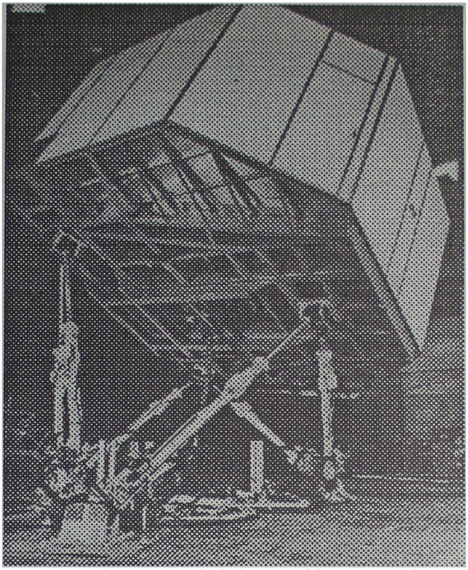


Figure 10. Link synergistic 6 degree-of-freedom motion platform -- a typical "six-post" configuration.

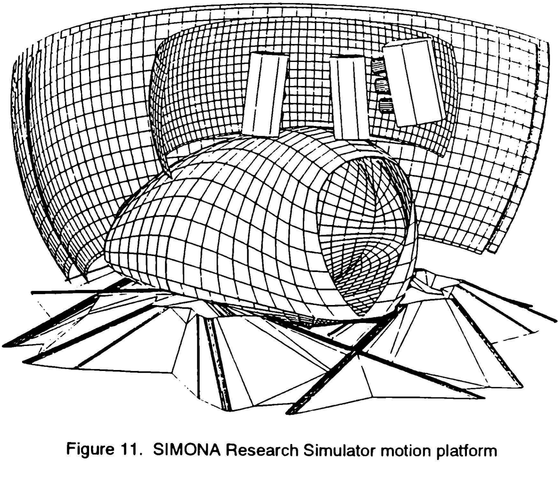


Figure 11. SIMONA Research Simulator motion platform

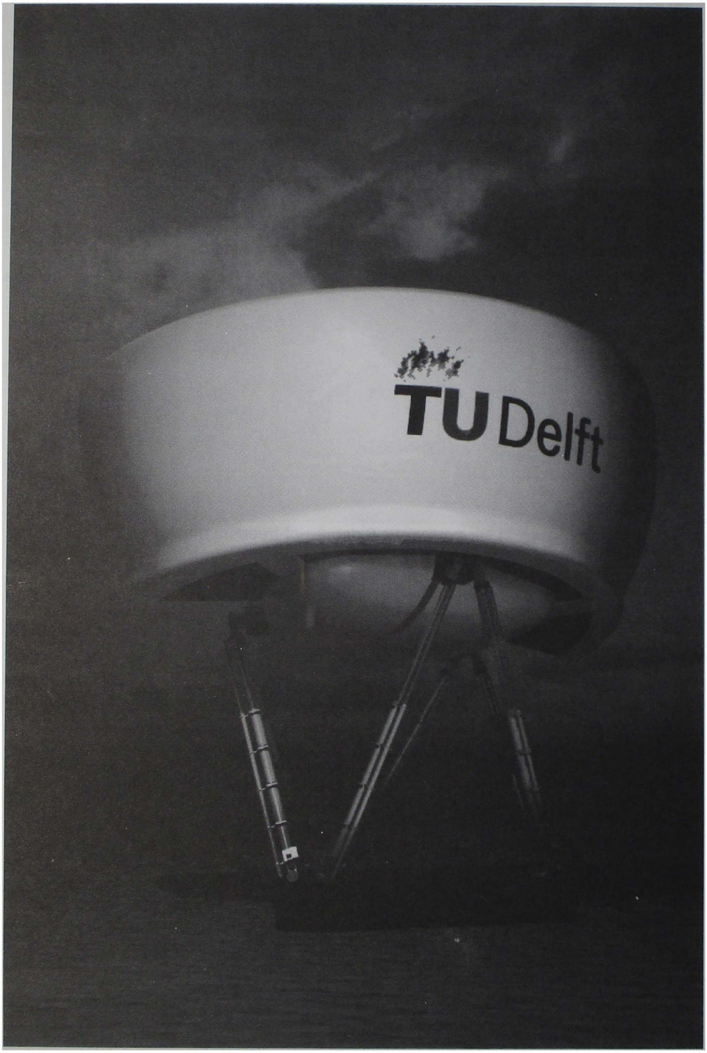


Figure 12. SIMONA Research Simulator (artist's concept)

AIAA-94-3419-CP

A PART-TASK SIMULATOR FOR ADVANCED AUTOMATION AND COMMUNICATIONS RESEARCH

Greg Pisanich, Ed Lee, Larry Beck

Sterling Software Inc.
Foster City, CA

NASA Ames Research Center
Flight Human Factors Research Branch
MS 262-6, Moffett Field, CA 94035-1000

Abstract

A part-task simulator to support research into advanced commercial flight deck and Air Traffic Control (ATC) automation and communication interfaces has been developed at the Flight Human Factors Research Branch (FLT) at NASA Ames. This system provides independent and integrated flight and ATC simulator stations, party line voice and datalink communications, along with video and audio monitoring and recording capabilities. Over the last several years, it has been used to support the investigation of human factors research issues involving: communication modality; message content and length; message formatting; and the graphical versus textual presentation of information. This paper describes this workstation-based simulator, including its goals, capabilities, development challenges, and future directions.

Introduction

A key goal of the Flight Human Factors Research Branch (FLT) at NASA Ames is to investigate the effect of new automation and communication interfaces on Air Traffic Control (ATC) and commercial flight deck personnel. This human-machine systems research is sponsored through NASA and FAA research initiatives including Aviation Safety and Automation (AS/A), Terminal Area Productivity (TAP), and High Speed Research (HSR). One aspect of this research is the investigation of advanced prototypes and concepts, which requires flexible and rapidly reconfigurable simulation capabilities.

A part-task simulator that provides this functionality has been developed within the FLT branch. This system provides separate flight and ATC simulator stations, party line vocal and datalink communications, and audio/video monitoring and recording capabilities. Over the last two years, and at various stages of development, it has been used to investigate human factors research issues involving: communication modality [1, 2, 3]; message content and length [4]; message formatting [5]; and the graphical versus textual presentation of information.

Motivation

There is a need to provide a range of flight simulation facilities to meet the requirements of basic research in advanced concepts, operational implementation issues, and integrated system development. The simulation facilities must be responsive to the level of fidelity required by the research agenda. Although high-fidelity, line oriented flight simulation is necessary to support some studies [6], hypotheses that focus on sub-system functions like those of communications in voice and datalink operations can be adequately examined in a part-task environment. Extensive scene generation and out the window cues, accurate handling qualities, and takeoff and landing operations are all issues on which automation and communication research currently does not focus.

There is also a need to provide shorter development cycles while also reducing costs. Part-task simulators promote a more hands on approach, in which rapid prototyping of interfaces and systems are possible and encouraged. These simulators normally require less overhead, including facility, personnel, and software development costs. The focused development possible with part-task simulators also allows shorter development schedules to be realized.

This technology also needs to be made more accessible. Delivering a part-task within a workstation environment provides the added benefit of portability and distributability. With instances of that part-task available to both internal researchers and outside grantees, a much higher level of collaborative work is achievable. If major module compatibility can be maintained, a direct upgrade path to the full motion simulator for that work can exist. With similar functionality, comparison of full-mission and part-task results becomes possible.

Goals

In developing the part-task simulator, we have attempted to meet the follow goals:

* Reconfigurable/reusable/distributable software.
* Software based on a homogeneous operating environment.
* Reasonable cost workstation architecture.
* Upward compatibility with full mission simulator.
*Hardware emulation in software for simplified interface reconfiguration.
* Common aircraft/avionics reference.
* End-to-end simulation facilities including ATC/flight deck integration.
* Network links to other facilities.
* Requirements of customers (researchers).

Part-Task Simulator

Facilities

The Part-Task simulation laboratory is located at the Flight Human Factors Branch at the NASA Ames Research Center in Moffett Field, CA. It consists of individual, reconfigurable rooms for flight deck, ATC, and experiment monitoring activities. The flight deck and ATC rooms are both sound-proofed and measure 8' x 8' each. The experiment monitoring activity occupies a 10' x 12' area and is partially enclosed. This room also serves as a video review area for post-experiment analysis.

Each room is equipped with wall and subfloor interconnects for computer networking, radio voice communications, and audio/video monitoring and recording. These interconnects allow the system to be customized to meet the specific requirements of each study. Figure 1. illustrates the physical layout of the laboratory along with the location of major facilities for an upcoming experiment.

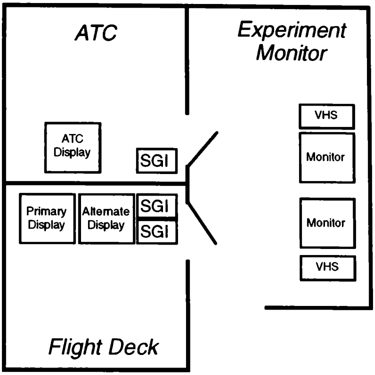


Figure 1. Physical Layout of Part-Task Laboratory.

System Architecture

Three Silicon Graphics XZ4000 Indigo workstations provide the computing power for the part-task laboratory. Each is configured with 64Mbs of RAM and a 1Gb. disk. These systems communicate via their own local ethernet subnet that can be isolated from the building network during simulations. Off-line development is provided by two similarly configured Indigo2 XL workstations. Figure 1. shows the current configuration of the lab with two of the workstations assigned to the flight deck and the third assigned to the ATC task.

All part-task software is developed in C, with FORTRAN used only when necessary to support already developed routines. The software architecture uses a communicating process, client/server methodology, which is supported using UNIX sockets. This architecture allows the processes to be distributed among the computers so that graphical and computational loads can be equalized for each system.

Flight Interface

Flight simulation capabilities are provided by an FLT rehost of the Crew-Vehicle Simulation Research Facility's (CVSRF) Advanced Concepts Flight Simulator (ACFS) [7] software to the Indigo workstation. This system provides a full glass cockpit interface consisting of primary flight instruments along with secondary synoptic displays as illustrated in Figure 2. The 747-400-style flight instruments include a primary flight (PFD) and weather-enhanced navigation display. The paged synoptic display provides access to various aircraft systems and a prototype datalink interface. The standardized architecture of the secondary display software allows new synoptics to be developed and integrated with a reduced amount of effort.

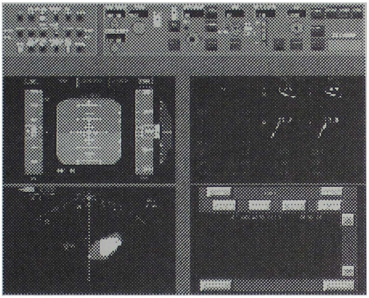


Figure 2. Primary and Secondary Flight Displays.

The flight simulator includes realistic aerodynamics and autoflight controls through a software based Mode Control Panel (MCP) and Flight Management System (FMS)/Control Display Unit (CDU). The FMS interface provides capabilities similar to a Boeing 757-class aircraft. The MCP resides on the main flight deck monitor (figure 2.), and the CDU is displayed on right side of the alternate monitor as illustrated in figure 3. The "soft" nature of the CDU and MCP interfaces have allowed many innovative automation concepts to be pursued that would be difficult to implement in a traditional full-mission simulator.

Aircraft performance and flight handling characteristics of the simulator are representative of a Boeing 757-class aircraft. While takeoff and landing capabilities exist, these phases are typically excluded from part-task scenarios. No "out the window" views are currently provided.

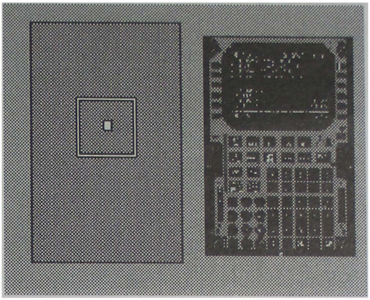


Figure 3. Secondary Task and CDU Displays.

Pilots interact with this system via mouse input and/or a BG-systems 3 degree of freedom joystick with integral button/lever box. This box allows the control of flaps and speedbrakes and also permits the use of alternative joysticks via an external interface. Direct manipulation of the flight interface will soon be available with the integration of Elotouch Systems' touch screen displays.

ATC

Monitoring and control of the aircraft through the airspace environment is accomplished through an ATC control/display simulator. This system is an enhancement of software originally developed by ARGO Systems for the CVSRF. It offers functionality beyond standard ATC displays including multiple resolution and repositioning capabilities. A secondary window is provided as a datalink message interface (see figure 4.).

Available airspace environments include Greater New York City and the San Francisco Bay Area to Los Angeles flight corridor. All major airports and NAVAIDs are defined with multiple ILS approaches provided at terminal airports. Enhancements to these as well as the development of new environments are provided as needed to support experiment requirements.

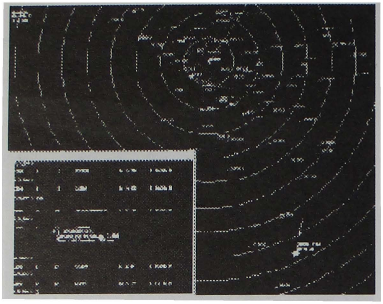


Figure 4. ATC Display.

Datalink

The Datalink communications system is an enhancement of full mission developed code with advanced capabilities for modification and format manipulation. The aircraft interface provides message accept, reject, and clear commands along with autoload capabilities into the MCP and/or CDU. Message log and review facilities are provided. A prototype interactive graphical datalink interface that can display steering commands on the PFD has also been developed. The ATC Datalink interface allows textual clearance and informational messages to be sent and monitored. Messages can be developed in advance and issued using keyboard commands or created on-the-fly as free text messages.

Secondary Task

The flight deck simulator provides the capability of measuring the difficulty of automation and communication tasks (such as datalink communication or CDU programming) by evaluating a pilot's performance at a concurrently performed "secondary" task. High performance at the secondary task indicates spare pilot attention capacity for the primary task. An earlier version of the secondary task involved monitoring a gauge for an out of range level. The data recorded for this discrete task included response time and error rates.

A continuous compensatory tracking task has been developed as the secondary task for an upcoming experiment. This task requires a pilot to use a joystick to maintain a cursor within a target area while a forcing function of random sine wave inputs attempts to drive it out. That target area is illustrated as the left side of figure 3. The deviation of the cursor from the center is collected as the tracking error, (secondary measurement).

The secondary task is integrated into the flight simulator with respect to UNIX process scheduling and data collection. This allows the secondary task to be run or tested independent of the flight simulator. A software framework for the secondary task has also been established, so that newly developed tasks can easily be substituted for the current instance.

Audio Communications

Verbal communication between the pilot and ATC is provided via a partyline communications interface. This system spans both rooms, supporting the monitoring and stereo (pilot/ATC) audio recording of verbal interactions for future review. Push To Talk (PTT) capability is provided to both the Pilot and ATC specialists.

To support the reproducibility between experimental runs and among controllers, the ATC specialist's voice commands can be digitized in advance and saved to disk using the Audio Interchange File Format (AIFF). These messages can be dispatched by the controller using keyboard commands at the appropriate time during the simulation. The computer outputs the message into the partyline as if it were spoken by the controller.

Data Collection

Data collected from the simulation is programmable and are written to disk as time, variable name, and value triplets. Both discrete and continuous variables are collected; examples include: FMC/CDU button presses, aircraft state data, and secondary task performance. Data collection is integrated into the process scheduler on a single computer, ensuring a consistent time reference.

Audio/Video Monitoring and Recording

The ability to monitor the actions and progress of the simulation subject is provided via the partyline audio and several color video inputs. A Super-VHS camera can be statically positioned within the subject room to collect physical movements and the overall flow of the experiment. Subject focus and exact manipulation of the interface can be collected via the direct output of the subject monitors (displays, changes, and mouse position). This capability is provided by Silicon Graphics' Gallileo video output hardware.

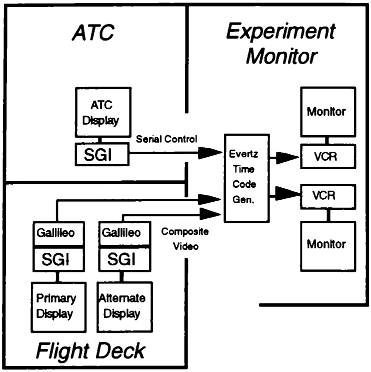


Figure 5. Part-Task Video Schematic.

This video can be routed to twin Super-VHS Video Cassette Recorders. The output from the audio communication system can be split and recorded to each of four available audio tracks. Time and subject information can be added to the tape via a programmable Evertz timecode generator. Figure 5 shows the video schematic for an upcoming experiment.

Development Challenges

The development of the part-task laboratory has not been without challenges. As with most projects, these have included both technical and institutional hurdles. The following paragraphs describe several issues we have encountered, not all of them with totally satisfactory solutions. This list would be helpful to anyone considering the development of a similar facility.

In order to speed development, and allow more experiments to be run (these are often conflicting goals), it is necessary to provide separate off-line development facilities. This eliminates contention for the system, and also allows testing to be performed concurrently with development. The extra cost should be traded off against shorter development cycles.

When software is adapted or rehosted from one system to another, any proprietary or copyright issues should be addressed well ahead of any planned development. Whether specific modules are proprietary or not, it can take months to determine their actual status for many reasons. Even if the developer and status is known, it can be difficult to track down the point of contact in a large company that can actually give the release.

Adapting a full motion simulator to the "small screen" of a workstation can also be a challenge. Software references to unneeded systems must be terminated properly for the simulation to run correctly. Some physical functionality can be easily duplicated using software displays and low cost hardware; for example, the flaps are implemented in the part-task using the levers on the BG-Systems box, but others, such as servo-driven autothrottles, are not that straightforward to reproduce.

Moving a system from one computer architecture to another creates special problems. The syntax of the Digital Equipment VAX and Silicon Graphics IRIX compilers are different; it would be desirable if this could be standardized. When rehosting software to a workstation environment, the resulting code should be made as distributable as possible (between computers). More than one workstation may be required to provide the necessary resources (instruction cycles, floating point operations, and screen real estate) to support the application.

Future Directions

Future development of the part-task lab will be driven largely by NASA flight deck research requirements associated with the TAP and HSR research initiatives. The goal is to continue to develop the base capabilities of the simulator with specialization performed as needed to support specific studies. Experiments planned for the near future include: an analysis of datalink message manipulation and negotiation interfaces, an investigation of pilot decision making using a new diagnostic interface, and a study of how pilot decision methodology is effected by time and informational constraints.

Enhancements to the flight deck systems will include a second pilot station to support crew interaction studies and the integration of additional aircraft systems, including "soft" representations of essential flight controls such as autothrottles, flaps, and speed brakes. Further development of the ATC environment will include the rehost of a version of the Center TRACON Advisory System (CTAS) [8] and a pseudo-pilot station to provide realistic background traffic and communications. Improvements to the simulation infrastructure will include dedicated high-speed network capabilities, the development of real-time data monitoring facilities, and the incorporation of a low-cost, flexible cockpit mockup.

References

1. A. McGann, S. Lozito, and K. Corker, *An Experimental Evaluation of Data Link Textual Formats,* (in press), San Jose State University Foundation, San Jose, CA, 1994.

2. D. Morrow, Part-task Study of Pilot-Controller Communications, Voice Communication Problem Results, *Subcontract Report 0342-6-240*, Sterling Software, Inc., Foster City, CA, 1994.

3. D. Morrow, M. Rodvold, Analyzing Problems In Routine Controller-Pilot Communication, *International Journal of Aviation Psychology*, 3, 385-302, 1993.

4. D. Morrow, M. Rodvold, *The Influence of ATC message length and Timing on Pilot Communication*, NASA contractor report 177621, NASA Ames Research Center, Moffett Field, CA, 1993.

5. A. McGann, S. Lozito, & K. Corker, *Cockpit Data Link Displays: An Evaluation of Textual Formats*, Presented at SAE AEROTECH, Anaheim, CA, 1992.

6. H. Orlady, R. Hennessy, R. Obermayer, D. Vreuls, and M. Murphy, *Using Full-Mission Simulation for Human Factors Research in Air Transportation Operations*, NASA Technical Memorandum 88330, NASA Ames Research Center, Moffett Field, CA, 1988.

7. Aerospace Human Factors Research Division, *Crew-Vehicle Systems Research Facility*, Information Bulletin, NASA Ames Research Center, Moffett Field, CA, 1994.

8. H. Erzberger. *CTAS: Computer Intelligence for Air Traffic Control in the Terminal Area*, NASA Technical Memorandum 103959, NASA Ames Research Center, Moffett Field, CA, 1992.

AIAA-94-3420-CP

Manned Flight Simulator and the Impact on Navy Weapons Systems Acquisition.

Roger A. Burton
Chad C. Miller
Rick E. Mills
SA100 / SY30
Flight Test and Engineering Group
Naval Air Warfare Center
Aircraft Division
Patuxent River, Maryland 20670
U.S.A.

ABSTRACT

The dramatic rise in the level of complexity and the cost of modern airborne weapons systems has overwhelmed the ability for conventional flight test techniques to evaluate system performance and specification compliance. The U.S. Navy has developed a unique, modern pilot-in-the-loop simulation facility targeted at reducing the development cost and shortening the time-line required for new aircraft systems. The Manned Flight Simulator facility, located at the Naval Air Warfare Center - Aircraft Division, provides a flexible simulation capability through the use of modular hardware and software designs that can handle almost all simulations required for Test and Evaluation and Training. This paper will survey technical capabilities, discuss applications to current major Navy projects and demonstrate how modularity and extensive use of shared assets reduce program costs, increases safety, and optimizes flight test. In addition, specific examples of technology transfer from the MFS to other government agencies and private industry will be explored.

LIST OF ABBREVIATIONS

DIS	Distributed Interactive Simulation
COTS	Commercial Off-The-Shelf
NAWC-AD	U.S. Navy's Naval Air Warfare Center - Aircraft Division
ACETEF	Air Combat Test and Evaluation Facility
T&E	Test and Evaluation
MFS	Manned Flight Simulator Facility
IOS	Instructor-Operator Station
IG	Image Generator

PROBLEM STATEMENT

The past decade has seen explosive growth in the complexity and cost of modern weapons systems. This presents a challenge for Test and Evaluation organizations, as the time allowed to test aircraft systems and sub-systems have not been expanding in a proportional fashion. Figure 1.0 shows this trend. Thus, T&E organizations have been forced to develop new methods for extensively testing new weapons systems yet remain within project time requirements.

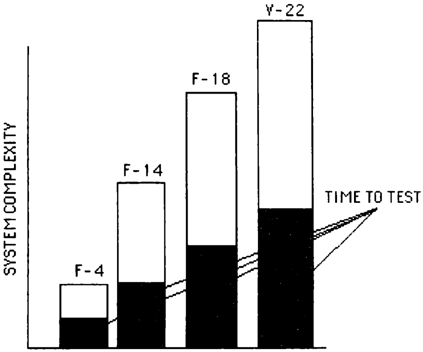


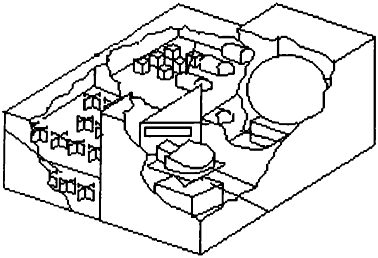
Figure 1.0

NAWC-AD, formerly the Naval Air Test Center, has developed the Manned Flight Simulator Facility to meet this challenge.

MANNED FLIGHT SIMULATOR OVERVIEW

The MFS is a single laboratory located in the Air Combat Test and Evaluation Facility (ACETEF).

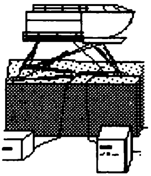
(Figure 2) The unique aspect of the MFS resides in the design of the simulation cockpits and interfaces at each of the facility simulation stations.



MFS FACILITY
Figure 2.0

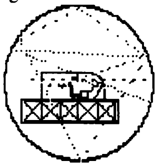
The MFS contains five simulation stations: a single 40 foot dome, a six degree of freedom motion base, and three engineering development stations. The facility has a central computer facility that contains a large variety of host computers and visual image generators. The central computer cluster includes Vax 6440's, Model 4000-90's, Silicon Graphics Onyx Racks with multiple reality engines and video output cards, ADI AD10's and AD100's and many other systems. For visual image generation the facility shares among the simulation stations a Compuscence IV, a Compuscence IVA and several ESIG-2000s. MFS is expecting delivery of a PT2000 in the near future. All of the IG'S output can be switched from one simulation station to the next as required with an in-house designed and constructed electronic video switcher.

The motion platform station features a Rediffusion motion base and Wide II infinity optical system that has a 200X45 field of view. The light-tight box has been modified to accept roll-in, roll-out cockpits.



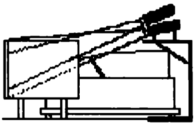
The motion station is capable of producing ±1.5G vertical ±0.7G lateral & longitudinal acceleration, 38.5° to +32° pitch with ±27.5° roll and ±36° yaw with ± 250 dpss angular acceleration. This station is primarily used for rotary wing applications and for the landing phase of fixed wing craft.

The forty foot dome has 360° x 290° low resolution background image with a 60° x 80° high resolution forward looking fixed insert.



The dome has two target projector pairs. It is used primarily where a large field of view and tactical maneuvering is required. Modular cockpit access is through large doors located at the second story entryway.

The three engineering development stations are used to support flight test and to develop and validate software development prior to production release of software.



The stations have a 165x40 display screen placed in front of the cockpits, and the screens are front illuminated from projectors mounted on overhead racks. One of the MFS lab stations is dedicated to a helmet mounted display system.

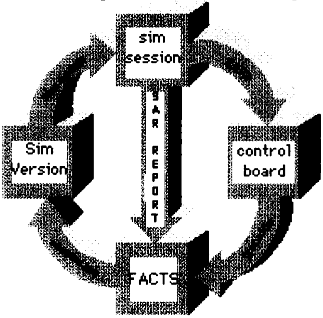
All of the MFS simulation stations are capable of running with actual aircraft flight hardware in the loop. Aircraft flight control and mission computers are located in the central resources cluster, and can be switched to the required simulation station as the cockpit modules move about. This capability is used extensively to validate avionics and flight control systems models, as well as perform testing on the aircraft equipment and software loads themselves.

STANDARD SIMULATION SOFTWARE

The MFS uses high fidelity, non-linear airframe models for its engineering support of T&E. Teams of engineers are assigned to each airframe with the charter to constantly update and improve the fidelity of these models, gathering data from any available source, and using advanced numerical techniques, such as Parameter Identification to update them. The wide range of simulations and software processes that the MFS has been required to simulate long ago highlighted the requirement for the facility to have a standard set of simulation executive software, easy to use and maintain, and applicable to all six-degree of freedom simulations. The Controls Analysis and Test Loop Environment (CASTLE) software executive was developed to meet this requirement. The system offers user interface windows, equations of motion, linear model extraction and parameter identification tools that work with any installed air vehicle simulation. To install a simulation in the MFS, the developer of the simulation provides the aerodynamic, engine and control system models, and plugs them into the shell software. The simulation can also be developed locally from flight test or wind tunnel data.

SOFTWARE DEVELOPMENT PROCESS

The MFS has set into place an extensive system for software discrepancy tracking and resolution. Called "FACTS" for Facility Anomaly Control and Tracking System, this system has allowed the MFS to retain tight control over the configurations of each simulation being used in the laboratory. The menu driven FACTS database is accessible at each engineers workstation over the facility network, and each engineer can review in real-time the status of work requests, discrepancy correction, etc. Each simulation is assigned version numbers that can be used to determine what changes have been made in the code since the previous version. The flow of this process illustrated in Figure 3.0.



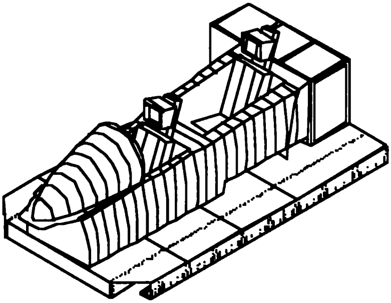
FACTS Flowchart
Figure 3.0

Simulation users generated System Anomaly Reports (SARs) during a lab session. These are entered into the FACTS database, and assigned to be corrected to an engineer or engineering team under the control of the configuration control leader. The work progress is also entered into the FACTS database to interested parties to track progress. When the problem is resolved, the simulation version number is updated to show that changes have been made, and the process begins again.

SIMULATION COCKPIT MODULES

Each simulation cockpit is of a Roll-In, Roll-Out (RIRO) design. The current "stable" of aircraft cockpits include the F-14, V-22, F-18A, F-18F, and AH-1W aircraft. The MFS also has a generic cockpit that can accommodate the grips and throttle clusters from any Navy or Marine aircraft. The cockpits all share a standard footprint base frame that includes mechanical interfaces for the motion base, and a common set of interface hardware to connect to the center host computer cluster.

This gives the MFS a powerful simulation capability in that whenever a new simulation is required, the cockpit can be constructed without requiring any simulation station to be disrupted. Typically, any facility installing a new simulation in an existing asset, be it a dome or motion base, would require months and perhaps years of down time to install and integrate the new simulation asset. At the MFS the new simulation software and the cockpit can be developed during use of the lab by other users, and integration of the new simulation capability, hardware and software, is not disruptive to the lab. The F/A-18F cockpit is shown in Figure 4.0. This example is typical of all the MFS designed and constructed cockpits.



F/A-18F Cockpit Module
Figure 4.0

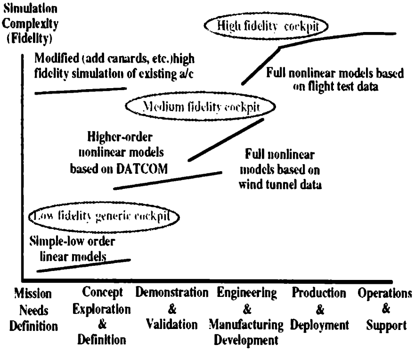
The F/A-18F cockpit was taken from an F/A-18B aircraft that was stricken from the Naval Register. The cockpit is equipped with 5 CTR type displays in both crew stations of a new design for this type aircraft. The use of actual aircraft equipment in the MFS is desired, but in the case of new and developing aircraft this may not always be practical. The cockpit uses high fidelity simulated equipment that has the same form fit and function as the actual aircraft equipment. An actual aircraft mission computer operating over 1553 buses drives these displays. The cockpit was designed to be able to use actual aircraft displays in a "pull-out plug-in" manner when they become available. The MFS standard VME-based interface is employed in the cockpit. The cockpit is motion capable for use in the MFS motion station, and contains digital electric control loaders to replicate the aircraft's force/feel to the pilot.

ACQUISITION SUPPORT - CONCEPT

The need for an MFS-type facility became apparent in the Navy during the EMD of the F-18 aircraft program. The size and complexity of the weapon system installed on that aircraft was a large step beyond any such system that the then-NATC had been asked to evaluate. To test out all possible configurations and malfunctions of the weapons system would have required years of testing, and still many test points would have been prohibited due to the high risk associated with various failure modes.

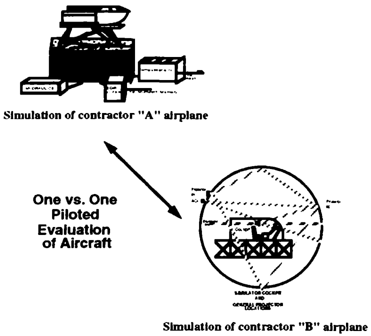
Application of Advanced Simulation and Analysis Capabilities are made to the Research, Development and Acquisition Process from Mission Needs Definition to Operations and Support. The concept of the availability of simulation during the Acquisition Process is shown in figure 5.

As shown, during the early part of the acquisition cycle high fidelity simulations of a specific vehicles/weapon systems are not available and simple lower order models and generic higher order models are used for analytical studies, design, mission analysis and performance predictions. As the development process proceeds and design goals and requirements are better defined, increasingly complex models and simulations become available to support the detailed design, development and testing process.


Simulation Complexity (Fidelity)
High fidelity cockpit
Modified (add canards, etc.)high fidelity simulation of existing a/c
Full nonlinear models based on flight test data
Medium fidelity cockpit
Higher-order nonlinear models based on DATCOM
Full nonlinear models based on wind tunnel data
Low fidelity generic cockpit
Simple-low order linear models
Mission Needs Definition
Concept Exploration & Definition
Demonstration & Validation
Engineering & Manufacturing Development
Production & Deployment
Operations & Support

Simulation Availability and Growth During the Acquisition Cycle
Figure 5

At the beginning phases of the Acquisition Process, many application of the MFS can be used to enhance the Navy's ability to participate in the development of a weapon system. During the Mission Needs Definition and Concept Exploration & Development phases, the utilization of the MFS in conjunction with ACETEF provides the pilot interface and computational resources to perform basic research and assess the needs of specific Navy missions requirements. For example, the MFS can be used to determine the operational requirements in a combat environment, determine the effects of aircraft aerodynamic configuration on operational capabilities and evaluate analysis methods for high angle of attack super maneuverability. Another example, would be to determine the effectiveness of a weapon system against current and projected threats. During Demonstration and Validation, the MFS provides the capability to do a Fly-off between two competitive aircraft configurations. During this Fly-off, an analytical evaluation and comparison of performance, stability and control, carrier suitability and weapons capabilities can be made to assess the two designs. This evaluation would include pilot evaluations of the two aircraft configurations to asses flying qualities, work load and weapon system effectiveness. A specific example of this would be to conduct piloted simulation carrier approaches and arrestments to evaluate of the carrier suitability of two or more competitive aircraft designs. This process is illustrated in figure 6.


Simulation of contractor "A" airplane
One vs. One Piloted Evaluation of Aircraft
Simulation of contractor "B" airplane

Utilization of MFS Motion Base and Dome for Evaluating Competing Designs
Figure 6

As the weapon system design completes Demonstration and Validation and the program enters into the Engineering Manufacturing and Development Phase, the MFS has an even bigger impact on system development and testing. Utilization of the MFS can be made to support aircraft development by enhancing the Navy's ability to follow and evaluate contractors designs. For example, in flight testing the Navy uses piloted simulation to plan flights, optimize flight profiles and practice high risk maneuvers and emergency procedures. The simulation is used as an analysis tool to understand complex systems such as FCS, make design trade-offs, and provides for efficient use of flight test time and assets.

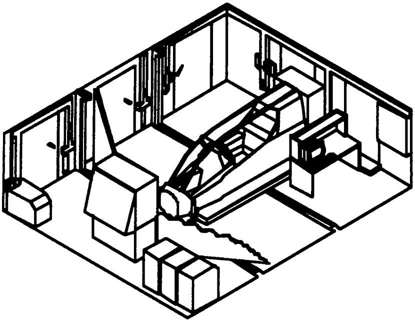
During the Production and Deployment and Operations and Support, the MFS provides for timely and independent analysis of fleet in-service support requirements and problems. This varies from mishap investigations to supporting the weapon system change process. These issues include, resolving competing requirements, evaluate ECP's, attending TCM, PDR, CDR meetings, supporting NATOPS changes, supporting ground and flight test and software life cycle requirements.

USE OF THE MFS FOR TRAINER SUPPORT

The concepts of the MFS software and hardware designs, and the high quality of the aerodynamic simulation databases developed has begun to transfer to the training community. The MFS has been involved in more and more programs that are outside of the its traditional role of pure test and evaluation support. The MFS has either directly updated, or delivered databases to the appropriate contractor to update existing Navy trainers. The MFS updated the F-14 training device 2F112 with an aerodynamic database, and performed and engine model upgrade on the F-14's 2F95 device. The F/A-18 high alpha departure database, and the full envelope AV-8B database have been provided to contractor agencies for installation into existing fleet assets.

The MFS hardware designs has also had impacts upon the trainer community. The MFS was tasked to produced a V-22 simulation of high enough quality that it could be used for the training of test pilots as well and T&E support. The MFS constructed a V-22 cockpit module that was an exact replica of V-22 aircraft number 3. and this device has been heavily used by the V-22 test team for both flight test support and training.

The MFS has also been involved in rapid prototyping for the trainer community. The AH-1W deployable Aircrew Procedures Trainer prototype was designed and constructed in-house by the MFS using our standard hardware and software. For the APT, however, our design team used new technology host computers that were small and rugged enough to be embedded into the cockpit module itself. The AH-1W APT rehosted the full Weapons Systems Trainer from the fleet training devices, and the entire device is constructed from off-the-shelf components. The ATP is shipped and sheltered in three Department of Defense mobile facilities, and replicates all functions of the WST except for motion cueing. The device was delivered 30 months after inception of the program. The device is pictured in the deployed configuration in Figure 5.0



AH-1W Mobile Procedures Trainer
Figure 5.0

This type of program highlights the flexibility and re-use capabilities of the MFS software and hardware standards.

Digital Flight Control System Acquisition Support Concept

DFCS Development Issues

With the introduction of the digital flight control system (DFCS) in tactical aircraft, the development and testing approach used for classical analog flight control systems were no longer sufficient and had to be modernized. In early and later DFCS programs, it was demonstrated that their development was time critical for meeting the overall aircraft schedules. The DFCS in these early programs was one of the critical aircraft systems that delayed certification for first flight. This situation existed because the DFCS is an extremely complex piece of flight critical hardware and software requiring a new approach for testing and development. These new development issues consisted of digital time delays which have a pronounced effect on aircraft flying qualities, extremely complex software development and redundancy management issues and integration requirements with other aircraft weapon systems.

DFCS Support Capacities Required

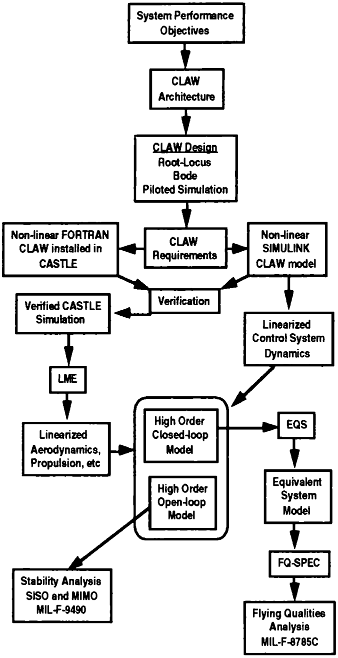
In response to the new approach needed for addressing the development issues for DFCS, the NAWCAD at Patuxent River, Maryland established a Flight Control System Support Activity including the required management structure and technical capabilities. In this capacity we execute the role as digital flight control subsystem software support activity, including life cycle planning for DFCS software. The analytical technical capabilities established included software verification and validation techniques, advanced control law design and stability analysis, redundancy management techniques and flight test methods. The hardware facilities established include manned flight simulation with cockpit and associated hardware (DDI, HUD, Display Processors, Stick, Throttle) and flight control computer test stations (FCCTS) which interface flight control computers/mission computers with the piloted simulation. A summary of our DFCS test capability is presented in figure 6.0.

Platform / Capability	F/A-18 A/B/C/D	F-14	V-22	EA-6B	F/A-18 E/F
High Fidelity Simulation	Yes	Yes	Yes	Yes	Yes
Pilot-in-the-Loop	Yes	Yes	Yes	Yes	6/96
FCC-in-the-Loop	Yes	6/94	5/94	6/94	1/95
Open Loop Testing	Yes	9/94	8/94	9/94	6/95
Closed Loop Testing	Yes	10/94	5/94	2/95	3/96

DFCS Test Capability
Figure 6.0

Integrated Control Law Design and Analysis

The Integrated control design and analysis capability is shown in figure 7.0. The approach integrates classical and modern control theory design concepts with simulation and automated analysis tools. As shown system performance objectives are established along with a control law architecture that addresses these requirements. The control law requirements are installed in non-linear form into our simulation architecture CASTLE and into our non-linear SIMULINK control law model. These control law implementations then go through a verification process. From the verified CASTLE simulation we use an automated linear model extraction tool to obtain a linearized aerodynamic and propulsion model. From the non-linear SIMULINK control law model we obtain linearized control system dynamics which are used to form a high order closed-loop model. An equivalent system analysis technique (EQS) developed in-house and is used to extract a low order model from the high order closed-loop model. This low order equivalent system model is used in conjunction with our automated flying qualities specification (FQ-SPEC) to perform a MIL-F-8785C flying qualities analysis. The CASTLE LME linearized models is used to form a high order open-loop models as the basis for conducting stability analysis to determine MIL F-9400 compliance for both SISO and MIMO characteristics.


System Performance Objectives
CLAW Architecture
CLAW Design
Root-Locus
Bode
Piloted Simulation
Non-linear FORTRAN CLAW installed in CASTLE
CLAW Requirements
Non-linear SIMULINK CLAW model
Verified CASTLE Simulation
Verification
Linearized Control System Dynamics
LME
Linearized Aerodynamics, Propulsion, etc
High Order Closed-loop Model
High Order Open-loop Model
EQS
Equivalent System Model
FQ-SPEC
Stability Analysis SISO and MIMO MIL-F-9490
Flying Qualities Analysis MIL-F-8785C

Integrated Control Law Design and Analysis
Figure 7.0

DFCS ACQUISITION SUPPORT EXAMPLES

F-18 and F-14 examples will be used to illustrate the DFCS acquisition support. In support of the F-18 DFCS, we have established a full six degree of freedom simulation and a flight control computer test bench. This capability has been used extensively to support DFCS development and flight test. For the F-18, we have more of a tradition role in supporting the DFCS design and development. We use our capabilities to provide detailed oversight of contractor development activities through simulation studies and analysis. However, when we receive a DFCS operational flight program update we conduct an independent assessment of DFCS and OFP performance using our F-18 flight control computer test station to conduct open loop test. We also perform closed loop test using our MFS pilot in the loop simulation coupled with the F-18 FCC's.

In the case of the F-14 our role is significantly more involved with the design effort. We lead a team of government, airframe contractor and flight control computer vendor engineers to develop a DFCS for the F-14. We have led the design effort using Navy and contractor engineers to design control laws and redundancy management. In this design effort, we used the control design approach discussed earlier coupled with evaluations of the design in our MFS piloted simulation. The concept of this program is also different in that two flight control test benches were built, one for the flight control vendor and one installed in the MFS. At the MFS we are currently using this flight control bench named the Engineering Test Set (ETS) to evaluate series actuator design and the previous analog flight control computer characteristics.

TECHNOLOGY TRANSFER AND THE MFS

Long before technology transfer became a requirement for DOD laboratories, designs and ideas from MFS were moving into private industry. The interchangeable high-roll stick design used in the MFS generic cockpit module has been patented and licensed to private industry, and the video switching IOS design used in the AH-1W APT has a patent pending.

Several commercial products now on the market have their origin in the MFS laboratory. The MFS itself builds and sells multiported memories that allow the connection of a large number of dissimilar computers to a common shared memory. MFS developed 1553 bus interfaces have been transferred to industry and are on the public market, along with emulation software for the Navy standard airborne computers. The digital electronic control loaders designed and constructed for use in the MFS, is now sold in the open market.

Engineers at MFS hold several Patents for the Navy, including the unique Instructor/Operator station developed for the AH-1W program and the reconfigurable high-roll stick developed for the generic simulation cockpit.

The CASTLE software has been adopted by the Canadian Armed Forces, the Australian Armed Forces, and several university systems.

SUMMARY

In summary, we currently have an operational MFS facility and trained Navy technical staff with a full spectrum of aircraft flight fidelity simulations and cockpits including the F-18 / F-14 / V-22 / EA-6B / X-31 / T-45 / AV-8B, etc. In addition to our airframe specific cockpits we have the MRC Cockpit available for evaluation of advanced aircraft configurations and advanced aircrew/avionics/control systems. We have an integrated Navy capability to support trainer prototyping and development. Our avionics and DFCS acquisition support capabilities are in place and include mission / FCC computer and other aircraft hardware integrated into the simulation environment. The F-18 and F-14 FCC Test Stations are operational and we have advanced analytical capabilities in the areas of flight control law and stability and control and redundancy management. The MFS is integrated with ACETEF for total weapons system development. We have demonstrated our capability to participate in networking (DIS) efforts through our involvement in HYDY, WAR BREAKER and MDT2 projects.

BIOGRAPHIES

Mr. Burton is Head of the Simulation and Control Technology Department of the Strike Aircraft Test Directorate at the Naval Air Warfare Center. He is responsible for managing and conducting research/development and acquisition support programs directed towards manned and computer simulations, modern control theory, advanced system identification technology, advanced flight control system test technology, flying qualities test techniques, and the evaluation of experimental and prototype aircraft. In conjunction with these programs Mr. Burton has worked with the NATF, A-12, F/A-18, AV-8B, F-14A, F-4S, E-2C, H-60, V-22, AD-1, X-29, S-3, and F-8 OWRA aircraft. Mr. Burton received a B.S and M.E. degree from Virginia Polytechnic Institute and State University in 1966 and 1970, respectively. He graduated from the U.S. Naval Test Pilot School in 1969.

Mr. Miller is currently the section head for the tactical aircraft simulation section (SA103) of the MFS facility and the Project Leader for the development of a AH-1W Trainer for the Marine Corp. He was the lead simulation engineer on the V-22 Government Test Pilot Trainer program between 1985 and 1990. He received his B.S. degree in Aerospace Engineering from North Carolina State University of Raleigh, North Carolina in 1986.

Mr. Mills is a section head in the Computer Technology and Simulation Department of the System Engineering Test Directorate at the Naval Air Warfare Center Aircraft Division. He has been Lab Manager for the MFS facility since 1986 and is responsible for the development, integration and infrastructure of the facility. He received his B.S. degree in Electrical Engineering from West Virginia Tech in 1984.

AIAA-94-3421-CP

A FEASIBILITY STUDY OF THE USE OF AXISYMMETRIC THRUST VECTORING FOR ENHANCED IN-FLIGHT SIMULATION

Dr. Dominick Andrisani, II* and Mark Alan Jones †
Purdue University, West Lafayette, Indiana 47907-1282

Abstract

The use of axisymmetric thrust vectoring technology to expand the VISTA (Variable Stability In-Flight Simulator Test Aircraft) F-16 simulation capability is explored. First, axisymmetric thrust vectoring control power is used to expand the VISTA Short Period and Dutch Roll Simulation Envelopes. This is accomplished by augmenting conventional aerodynamic control surface control power with axisymmetric thrust vectoring control power. The increased control power is used to respond to feedback command signals to expand the number of Short Period and Dutch Roll natural frequency and damping ratio combinations the VISTA can simulate at fixed flight conditions. Second, axisymmetric thrust vectoring control power is used to replace the aerodynamic control power required for high fidelity VISTA tracking. Model Following Flight Control Systems (MFFCS) are developed for the VISTA with and without thrust vectoring. In-flight simulation tracking operations are performed for both VISTA configurations. VISTA aerodynamic control surface effort during in-flight simulation is measured for each configuration. Benefits provided by axisymmetric thrust vectoring are quantified in terms of percent reduction in peak and RSS (root of the summed square) aerodynamic control effort for the simulation.

Nomenclature

A	= VISTA F-16 rigid body state matrix
$\bar{A}$	= VISTA F-16 full order longitudinal state matrix
A_m	= HARV F-18 rigid body state matrix
$\bar{A}_m$	= HARV F-18 full order longitudinal state matrix
B	= VISTA F-16 rigid body control matrix
$\bar{B}$	= VISTA F-16 full order long. control matrix
B_m	= HARV F-18 rigid body control matrix
$\bar{B}_m$	= HARV F-18 full order long. control matrix
$\bar{F}$	= VISTA F-16 full order lat/dir state matrix
$\bar{F}_m$	= HARV F-18 full order lat/dir state matrix
$\bar{G}$	= VISTA F-16 full order lat/dir control matrix
$\bar{G}_m$	= HARV F-18 full order lat/dir control matrix
J_{lat}	= lateral/directional performance index
J_{lon}	= longitudinal performance index
M	= pseudo control mixing matrix
RSS	= root summed square measurement
p	= body axis roll rate, deg/s
q	= body axis pitch rate, deg/s
r	= body axis yaw rate, deg/s
u	= forward perturbation velocity, ft/s
u	= VISTA F-16 rigid body control vector
u_m	= HARV F-18 rigid body control vector
u_{lat}	= lateral/directional full order control vector
u_{lon}	= longitudinal full order control vector
x	= VISTA F-16 rigid body state vector
x_{lat}	= lateral/directional full order state vector
x_{lon}	= longitudinal full order state vector
x_m	= HARV F-18 state vector
α	= angle of attack, deg
β	= angle of sideslip, deg
ϕ	= roll attitude angle, deg
θ	= pitch attitude angle, deg
σ_1	= longitudinal pseudo control mixing ratio
σ_2	= lateral/directional pseudo control mixing ratio
δ_a	= aileron deflection, deg
δ_f	= flap deflection, deg
δ_{df}	= differential flap deflection, deg
δ_h	= stabilator deflection, deg
δ_{dh}	= differential stabilator deflection, deg
δ_m	= longitudinal pseudo control deflection, deg
δ_n	= lateral/directional pseudo control deflection, deg
δ_r	= rudder deflection, deg
δ_T	= engine thrust level, pounds
δ_y	= yaw thrust vectoring deflection, deg
δ_z	= pitch thrust vectoring deflection, deg

* Associate Professor, School of Aeronautics and Astronautics, Senior Member AIAA.
† Graduate Student, School of Aeronautics and Astronautics.

Introduction

The United States Air Force (USAF) operates variable stability in-flight simulator test aircraft in aeronautical research and development projects. The primary use for the variable stability aircraft is to simulate, while airborne, the flight characteristics and human interfaces of current and future aircraft. The USAF has operated the NT-33A as the primary fighter variable stability aircraft since 1957. During its use, the NT-33A has supported the development of other United States air vehicles, such as A-10, X-15, F-15, NF-16 and the AFTI F-16. However, the age of the NT-33A has become a concern. It is the oldest flying aircraft in the USAF inventory, and is becoming both difficult and expensive to maintain. In addition, the performance of the NT-33A no longer represents performance of modern day fighter aircraft.

In response, USAF has overseen the development of a new fighter variable stability in-flight simulator. The new aircraft is called the VISTA F-16. VISTA is based upon the production F-16D employed by the Israeli Defense Forces. VISTA deletes the F-16D gun, ammunition drum, and self defense equipment, and also has an increased capacity hydraulics. No additional sources of aerodynamic or propulsive control power beyond those of the F-16D are found on VISTA.

Axisymmetric Thrust Vectoring

The advantage of using thrust vectoring as an additional source of control power in those regions of the flight envelope where aerodynamic control surfaces are less effective is understood. Until recently, it was difficult and expensive to add thrust vectoring capability to an existing aircraft. The development of axisymmetric thrust vectoring engine nozzles (by both General Electric and Pratt and Whitney) that can be easily retrofitted to existing aircraft now makes the job of modifying an aircraft with thrust vectoring capability more feasible.

The ability to quickly retrofit an aircraft with thrust vectoring capability has generated interest in using axisymmetric thrust vectoring on VISTA. VISTA simulation capability is directly limited by total available control power. The addition of axisymmetric thrust vectoring provides significant pitching and yawing moment control power to complement the VISTA aerodynamic flight controls. An increase in total available control power resulting from use of thrust vectoring clearly would expand the in-flight simulation performance of VISTA.

Feasibility Study Goals

The feasibility study generates preliminary results which quantify the expansion of VISTA simulation ability resulting from use of axisymmetric thrust vectoring control technology during in-flight simulation.[1] Expansion of VISTA simulation capability is explored in terms of:

- Growth of Short Period mode and Dutch Roll mode simulation envelopes. The growth of the simulation envelopes is a result of augmenting the aerodynamic flight control power used to respond to feedback command signals with thrust vectoring control power,

- Reduction in the aerodynamic flight control activity throughout in-flight simulation tracking. The reduction of control activity results from replacing the aerodynamic control power required for high fidelity simulation with thrust vectoring control power.

Reduced control surface activity allows VISTA to maintain high fidelity in-flight simulation of target aircraft responses to increased amplitude commands without saturation of the aerodynamic flight controls.

VISTA F-16 Linear Models

Linear models are developed which represent the unaugmented VISTA longitudinal and lateral/directional dynamics for a straight, symmetrical, wings-level, power approach, trimmed flight condition. Model data for rigid body dynamics and actuator dynamics are taken from the VISTA F-16 Engineering Analysis Simulation (EAS) maintained by the VISTA F-16 Advanced Development Program Office (ADPO) at Wright-Patterson Air Force Base. Actuator models include position and rate limits for each control.

The longitudinal linear model is represented in state space form as

$$\{x'_{\mathrm{lon}}\} = [\overline{A}]\{x_{\mathrm{lon}}\} + [\overline{B}]\{u_{\mathrm{lon}}\}. \qquad (1)$$

The prime denotes here differentiation with respect to time. Included in the longitudinal state space relationship for VISTA with thrust vectoring are rigid body dynamics (fourth order), stabilator actuator dynamics (fourth order), trailing edge flaps actuator dynamics (fourth order), thrust level dynamics (first order) and pitch thrust vectoring actuator dynamics (second order).

The lateral/directional linear model is represented in state space form as

$$\{x'_{lat}\} = [\overline{F}]\{x_{lat}\} + [\overline{G}]\{u_{lat}\}. \qquad (2)$$

Included in the lateral/directional state space relationship for VISTA with thrust vectoring are rigid body dynamics (fourth order), differential stabilator actuator dynamics (fourth order), differential trailing edge flaps actuator dynamics (fourth order), rudder actuator dynamics (fourth order) and yaw thrust vectoring actuator dynamics (second order).

Target Aircraft (HARV F-18) Linear Models

Linear models are developed which represent the target aircraft (HARV F-18) longitudinal and lateral/directional dynamics for a straight, symmetrical, wings-level, power approach, trimmed flight condition. HARV rigid body dynamics are taken from linear models cited by Sperry Corporation.[2] HARV actuator dynamics are from linear models cited by NASA.[3] Actuator position and rate limits are included in the models for each control.

The HARV longitudinal linear model is represented in state space form as

$$\{x'_{lon}\} = [\overline{A}_m]\{x_{lon}\} + [\overline{B}_m]\{u_{lon}\}. \qquad (3)$$

Included in the longitudinal state space relationship are HARV rigid body dynamics (fourth order), stabilator actuator dynamics (fourth order), trailing edge flaps actuator dynamics (second order) and pitch thrust vectoring actuator dynamics (second order).

The HARV lateral/directional linear model is represented in state space form as

$$\{x'_{lat}\} = [\overline{F}_m]\{x_{lat}\} + [\overline{G}_m]\{u_{lat}\}. \qquad (4)$$

Included in the lateral/directional state space relationship are HARV rigid body dynamics (fourth order), differential stabilator actuator dynamics (fourth order), aileron actuator dynamics (second order) and rudder actuator dynamics (second order).

Rigid Body Mode Simulation Envelopes

The ability of VISTA to simulate target aircraft rigid body modes with varying values of natural frequency and damping ratio is explored.[4] A highly effective but simple manner in which to manipulate the values of rigid body mode natural frequency and damping ratio is used.

Short Period Simulation Envelopes

It is known that Short Period natural frequency is heavily dependent upon α-to-pitching-moment-control feedback gains. Short Period damping ratio is minimally dependent upon this feedback gain. It is also known that both Short Period natural frequency and damping ratio are moderately dependent upon q- to-pitching-moment-control feedback gains. A specified pair of Short Period natural frequency and damping ratio is attained with proper choice of the α and q feedback gain values.

The ability to attain specified VISTA Short Period natural frequency and damping ratio indicates that the Short Period simulation envelope is determined by calculating values of natural frequency and damping ratio at all allowable combinations of α and q feedback gains to the pitching moment control. The maximum and minimum values which the feedback gains can assume define the Short Period simulation envelope boundaries.

Consider VISTA employing α and q feedback to the stabilators and pitch thrust vectoring where unaugmented VISTA longitudinal dynamics are defined in (1). At fixed values of feedback gains, the α-to-stabilator-deflection-command transfer function

$$\frac{\alpha(s)}{\delta_{h_c}(s)} = \frac{N_\alpha(s)}{D_\alpha(s)} \qquad (5)$$

is established. The frequency response gain data, $G_h(j\omega)$, and phase data, $\varphi_h(j\omega)$, are calculated at frequency ω_i, where ω_i is one of n logarithmically spaced frequencies between 0.3 radians per second and 10 radians per second.

A low order equivalent α-to-stabilator-deflection-command transfer function of the form

$$\frac{\alpha(s)}{\delta_{h_c}(s)} = \frac{K\omega_n^2 e^{-\tau s}}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad (6)$$

is determined. The free parameters in the equivalent system are steady state gain, K, natural frequency, ω_n, damping ratio, ζ, and equivalent time delay, τ. Parameter values are chosen to yield an equivalent system which minimizes the cost function, J_{lon}, defined as

$$J_{lon} = \frac{20}{n}\sum_{\omega_1}^{\omega_n}[(G_h - G_l)^2 + 0.01745(\varphi_h - \varphi_l)^2]. \qquad (7)$$

$G_l(j\omega)$ is equivalent system gain data and $\varphi_l(j\omega)$ is equivalent system phase data, both calculated at frequency ω_i. Low order equivalent system use is allowed for high order aircraft in the military standard for flying qualities MIL-STD 1797A.[5]

With the equivalent system established the values of natural frequency, ω_n, and damping ratio, ζ, are collected. Since this analysis is performed at a fixed set of feedback gain values, B2, B3, B14 and B15, those values of natural frequency and damping ratio represent only one point on the Short Period simulation envelope.

The Short Period simulation envelope for the VISTA without thrust vectoring is established by maintaining feedback gains B14 and B15 fixed at zero, representing no use of pitch thrust vectoring in feedback. Feedback gains B2 (α to δ_h) and B3 (q to δ_h) are incremented between maximum values and minimum values. Maximum values for B2 and B3 are 6.0 degrees per degree and 0.6 degrees per degree per second, respectively. Minimum values for B2 and B3 are fixed by Short Period mode stability requirements. At each fixed value set for B2 and B3, the equivalent system analysis described above is performed. Resulting values of equivalent system natural frequency and damping ratio are plotted.

The Short Period simulation envelope for the VISTA with thrust vectoring is established by incrementing feedback gains B2, B3, B14 (α to δ_z) and B15 (q to δ_z) between maximum values and minimum values. Maximum values for B14 and B15 are 6.0 degrees per degree and 0.6 degrees per degree per second, respectively. Minimum values for B14 and B15 are fixed by Short Period mode stability requirements. At each fixed value set for B2, B3, B14 and B15, the equivalent system analysis described above is performed. Resulting values of equivalent system natural frequency and damping ratio are plotted.

The expansion of Short Period simulation envelope resulting from the addition of thrust vectoring is clearly demonstrated by overlaying the envelopes as shown in Figure 1.

Dutch Roll Simulation Envelopes

It is known that Dutch Roll natural frequency is heavily dependent upon β-to-yawing-moment-control feedback gains. It is also known that Dutch Roll damping ratio is heavily dependent upon r-to-yawing-moment-control feedback gains. A specified pair of Dutch Roll natural frequency and damping ratio values is attained with proper choice of the β and r feedback gains.

The ability to attain specified VISTA Dutch Roll natural frequency and damping ratio indicates that the Dutch Roll simulation envelope is determined by calculating values of natural frequency and damping ratio at all allowable combinations of β and r feedback gains to the yawing moment control. The maximum and minimum values which the feedback gains can assume define the Dutch Roll simulation envelope boundaries.

Consider VISTA employing β and r feedback to the rudder and yaw thrust vectoring where unaugmented VISTA lateral/directional dynamics are defined in (2). At fixed values of feedback gains, the β-to-rudder-deflection-command transfer function and the ϕ-to-differential flaps-deflection-command transfer function

$$\frac{\beta(s)}{\delta_{r_c}(s)} = \frac{N_\beta(s)}{D_\beta(s)} \qquad (8)$$

$$\frac{\phi(s)}{\delta_{df_c}(s)} = \frac{N_\phi(s)}{D_\phi(s)} \qquad (9)$$

are established. The frequency response gain data and phase data are calculated for both transfer functions at frequency ω_i, where ω_i is one of n logarithmically spaced frequencies between 0.1 radians per second and 10 radians per second.

A low order equivalent β-to-rudder-deflection-command system and a low order equivalent ϕ-to-differential flaps-deflection-command system of the form

$$\frac{\beta(s)}{\delta_{r_c}(s)} = K_\beta \frac{(s+\tau_{\beta_1})(s+\tau_{\beta_2})(s+\tau_{\beta_3})e^{-\tau_\beta s}}{(s+\frac{1}{\tau_s})(s+\frac{1}{\tau_r})(s^2+2\omega_{n_{DR}}\zeta s+\omega_{n_{DR}}^2)} \qquad (10)$$

$$\frac{\phi(s)}{\delta_{df_c}(s)} = K_\phi \frac{(s^2+\tau_{\phi_1}s+\tau_{\phi_2})e^{-\tau_\phi s}}{(s+\frac{1}{\tau_s})(s+\frac{1}{\tau_r})(s^2+2\omega_{n_{DR}}\zeta s+\omega_{n_{DR}}^2)} \qquad (11)$$

are determined. The gains K_β and K_ϕ are defined as

$$K_\beta = \frac{K_1 \omega_{n_{DR}}^2 \frac{1}{\tau_s} \frac{1}{\tau_r}}{\tau_{\beta_1} \tau_{\beta_2} \tau_{\beta_3}} \text{ and } K_\phi = \frac{K_2 \omega_{n_{DR}}^2 \frac{1}{\tau_s} \frac{1}{\tau_r}}{\tau_{\phi_2}}.$$

The free parameters in the equivalent systems are steady state gain, K_1 and K_2, Dutch Roll natural frequency, ω_n, Dutch Roll damping ratio, ζ, Roll mode time constant, τ_R, Spiral mode time constant, τ_S, as well as equivalent system zero locations. Parameter values are chosen to yield the two equivalent systems which minimize the cost function, J_{lat}, defined as shown below

$$J_{lat} = \frac{20}{n} \sum_{\omega_1}^{\omega_n} [(G_{h\beta} - G_{l\beta})^2 + (G_{h\phi} - G_{l\phi})^2$$

$$+ 0.01745\,[(\varphi_{h\beta} - \varphi_{l\beta})^2 + (\varphi_{h\phi} - \varphi_{l\phi})^2)]. \tag{12}$$

$G_{l\beta}(j\omega)$ and $\varphi_{l\beta}(j\omega)$ are the β-to-rudder-deflection-command equivalent system transfer function gain and phase data, respectively. $G_{l\phi}(j\omega)$ and $\varphi_{l\phi}(j\omega)$ are the ϕ-to-differential flaps-deflection-command equivalent system gain and phase data, respectively, all calculated at frequency ω_i.

From the established equivalent systems the values of Dutch Roll natural frequency, ω_n, and damping ratio, ζ, are collected. Again, since this analysis is performed at a fixed set of feedback gain values, B25, B27, B29 and B31, those values of equivalent system natural frequency and damping ratio represent only one point on the Dutch Roll simulation envelope.

The Dutch Roll simulation envelope for the VISTA without thrust vectoring is established by maintaining feedback gains B29 and B31 fixed at zero, representing no use of yaw thrust vectoring in feedback. Feedback gains B25 (β to δ_r) and B27 (r to δ_r) are incremented between the minimum values and maximum values. Minimum values of B25 and B27 are -6.0 degrees per degree and -0.6 degrees per degree per second, respectively. Maximum value of B25 is fixed by the appearance of a roll-spiral oscillatory mode. Maximum value of B27 is 0.6 degrees per degree per second. At each fixed value set for B25 and B27, the equivalent system analysis described above is performed. Resulting values of equivalent system natural frequency and damping ratio are plotted.

The Dutch Roll simulation envelope for VISTA with thrust vectoring is established by incrementing feedback gains B25, B27, B29 (β to δ_y) and B31 (r to δ_y) between minimum and maximum values. Minimum values for B29 and B31 are -6.0 degrees per degree and -0.6 degrees per degree per second, respectively. Maximum value for B29 is fixed by the appearance of a roll-spiral oscillatory mode. Maximum value for B31 is 0.6 degrees per degree per second. At each fixed value set for B25, B27, B29 and B31, the equivalent system analysis described above is performed. Resulting values of equivalent system natural frequency and damping ratio are plotted.

The expansion of Dutch Roll simulation envelope resulting from the addition of thrust vectoring is clearly demonstrated by overlaying the envelopes as shown in Figure 2.

Model Following Flight Control Systems

A variable stability model following flight control system (MFFCS) is developed to alter the VISTA flight characteristics to yield high fidelity tracking of the target aircraft response during in-flight simulation. Altering VISTA flight characteristics is accomplished by commanding feedforward and feedback flight control commands. Feedforward control commands are linear combinations of target aircraft commands. Feedback control commands are linear combinations of measured VISTA responses.

The MFFCS consists of feedback, feedforward and command acceleration subsystems. The first MFFCS mathematical objective is to achieve the minimal difference between VISTA and target aircraft state vectors. The second MFFCS mathematical objective is to achieve minimal difference between VISTA and target aircraft state derivative vectors throughout simulation. The mathematical objectives are written as:

$$x = x_m \tag{13}$$

$$x' = x'_m. \tag{14}$$

Consider general linear models representing VISTA and HARV rigid body dynamics, but containing no actuator dynamics. The general linear models are written as:

$$\text{VISTA:} \quad \{x'\} = [A]\{x\} + [B]\{u\} \tag{15}$$

$$\text{HARV:} \quad \{x'_m\} = [A_m]\{x_m\} + [B_m]\{u_m\}. \tag{16}$$

Define a pseudo control vector, $\{u_p\}$, for VISTA. The pseudo control vector establishes a set of fictitious controls that are linear combinations of VISTA controls. Linear combinations are defined by the pseudo control mixing matrix, M

$$\{u\} = [M]\{u_p\}. \tag{17}$$

Mixing VISTA controls into pseudo controls accommodates the exchange of redundant control power provided by the stabilator and pitch thrust vectoring, and also the rudder and yaw thrust vectoring. For the thrust vectoring capable VISTA F-16 longitudinal MFFCS, the pseudo control mixing matrix, M_1, is represented in block diagram form in Figure 3. Pseudo control mixing ratio, σ_1, is constrained to values between zero and one. $(1-\sigma_1)$ degrees of stabilator deflection is commanded per degree pitching moment pseudo control deflection, δ_m. (σ_1) degrees of pitch thrust vectoring deflection is commanded per degree of pitching moment pseudo control deflection.

For the thrust vectoring capable VISTA F-16 lateral/directional MFFCS, the pseudo control mixing matrix, M_2, is represented in block diagram form in Figure 4. Pseudo control mixing ratio, σ_2, is constrained to values between zero and one. $(1-\sigma_2)$ degrees of rudder deflection is commanded per degree of yawing moment pseudo control deflection, δ_n. (σ_2) degrees of yaw thrust vectoring deflection is commanded per degree of yawing moment pseudo control deflection.

Substituting (17) into (15) yields a modified VISTA state space relationship in terms of the pseudo controls

$$\{x'\} = [A]\{x\} + [BM]\{u_p\}. \tag{18}$$

Application of the MFFCS mathematical objectives to the linear model for HARV in (16) and the modified linear model for VISTA in (18) yields the MFFCS control law in terms of pseudo controls

$$u_p = [(BM)^+(A_m - A)]\{x\} + [(BM)^+B_m]\{u_m\}. \tag{19}$$

Note that $[BM]^+$ represents the pseudo inverse of the matrix product [BM]. The MFFCS control law in (19) is transformed into a control law in terms of VISTA controls by demixing the pseudo controls, as shown in (17). The final form of the MFFCS control law is written:

$$u = [M(BM)^+(A_m - A)]\{x\} + [M(BM)^+B_m]\{u_m\}. \tag{20}$$

VISTA F-16 Simulation of HARV F-18 Responses

The MFFCS developed above are applied to an in-flight simulation tracking problem. Consider the VISTA and HARV trimmed in straight, symmetrical, wings-level, power approach flight conditions defined below:

VISTA:	Altitude	3200 feet
	Velocity:	283 feet per second
	Mach:	0.236
	α:	11 degrees
HARV:	Altitude	5000 feet
	Velocity:	299 feet per second
	Mach:	0.273
	α:	10 degrees

Linear models of VISTA and HARV are generated and used to develop MFFCS control laws as previously discussed. Longitudinal MFFCS allows VISTA to simulate HARV rigid body responses to longitudinal flight control commands. Lateral/directional MFFCS allows VISTA to simulate HARV rigid body responses to lateral/directional flight control commands.

VISTA in-flight simulations of HARV responses to a stabilator doublet command are performed at pseudo control mixing ratio values, σ_1, ranging from zero (i.e. no use of pitch thrust vectoring) to one (i.e. no use of stabilator).

VISTA in-flight simulations of HARV responses to a rudder doublet command are performed at pseudo control mixing ratio values, σ_2, ranging from zero (i.e. no use of yaw thrust vectoring) to one (i.e. no use of rudder).

Summary of Thrust Vectoring Benefits

The benefits provided by axisymmetric thrust vectoring to VISTA during in-flight simulation tracking are quantified by the reduction of RSS and peak aerodynamic flight control activity for the duration of the simulation. RSS aerodynamic flight control activity is defined

$$\delta_{RSS} = \sqrt{\frac{1}{n}\ [\sum_{i=1}^{n}(\delta(i))^2]} \tag{21}$$

where $\delta(i)$ is the flight control deflection at n discrete times linearly spaced through the simulation. RSS and peak stabilator activity are shown in Table 1 for VISTA longitudinal simulations with pseudo control mixing ratio values of σ_1=0.00 and σ_1=0.75. RSS and peak differential stabilator, differential flaps and rudder activities are shown in Tables 2 through 4 for VISTA lateral/directional simulation with pseudo control mixing ratio values of σ_2=0.00 and σ_2=0.90.

Table 1 Stabilator RSS and Peak Activity Comparison

	σ = 0.00	σ = 0.75	% Reduction
RSS Activity	0.1420°	0.0758°	45
Peak Activity	1.3314°	0.7207°	44

Table 2 Diff. Stabilator RSS and Peak Activity Comparison

	σ = 0.00	σ = 0.90	% Reduction
RSS Activity	4.3461°	0.6122°	85
Peak Activity	9.5584°	1.0777°	88

Table 3 Diff. Flaps RSS and Peak Activity Comparison

	σ = 0.00	σ = 0.90	% Reduction
RSS Activity	3.6600°	0.5310°	85
Peak Activity	8.1129°	0.9139°	88

Table 4 Rudder RSS and Peak Activity Comparison

	σ = 0.00	σ = 0.90	% Reduction
RSS Activity	3.6306°	0.0266°	99
Peak Activity	7.7414°	0.0559°	99

The use of axisymmetric thrust vectoring during VISTA in-flight simulation significantly reduces the RSS and peak aerodynamic flight control activities. Exact percent reductions of RSS and peak aerodynamic flight control activity are shown in the above tables for the nominal values of pseudo control mixing ratios.

Aerodynamic and propulsive flight control peak deflections for longitudinal and lateral/directional VISTA simulations are shown as functions of pseudo control mixing ratio, σ_1 or σ_2. Peak stabilator deflection and peak pitch thrust vectoring deflection are shown in Figure 5 and Figure 6, respectively, as functions of σ_1. Peak differential stabilator deflection, peak differential flaps deflection, peak rudder deflection and peak yaw thrust vectoring deflection are shown in Figures 7 through 10, respectively, as functions of σ_2. These figures demonstrate the capability of the flight control system designer to specify the amount of aerodynamic control effort reduction with proper choice of the pseudo control mixing ratio values in the flight control law.

Conclusions

A feasibility study is detailed quantifying the benefits axisymmetric thrust vectoring provides to VISTA for improved in-flight simulation capability. The VISTA rigid body mode simulation capability is studied with and without thrust vectoring in the feedback path. Feeding back VISTA response to thrust vectoring significantly expands the VISTA Short Period and Dutch Roll simulation envelopes in a power approach flight condition.

A model following flight control law is developed which alters VISTA flight characteristics to yield accurate simulation of the target aircraft, HARV F-18. In-flight simulations by VISTA employing the MFFCS are run at various values of pseudo control mixing ratios. Pseudo control mixing ratios fix the degree of thrust vectoring use. Aerodynamic flight control surface activity is observed during the simulations to determine the advantages provided by thrust vectoring during tracking. The reductions in aerodynamic flight control surface RSS activity and peak activity allow VISTA to simulate model aircraft responses to increasing amplitude commands without surface saturation. Since saturation of the aerodynamic flight control surfaces is one key factor which bounds the envelope within which VISTA accurately simulates its model aircraft, it is clear that use of axisymmetric thrust vectoring expands the VISTA in-flight simulation tracking capability.

References

[1] Jones, M. A., "A Feasibility Study of the Use of Axisymmetric Thrust Vectoring for Enhanced In-flight Simulation", Purdue University, December 1993.

[2] Sperry Corporation, F/A-18 Trim Cases, Linear Modeling and Time Responses for SP226, November 14, 1986.

[3] Buttrill, C.S., Arbuckle, P.D., and Hoffler, K.D., "Simulation Model of a Twin-Tail, High Performance Airplane.", NASA TM 107601, July 1992.

[4] Nguyen, B.T., Hamilton-Jones, L.T., and Leggett, D.B., "Analysis of the VISTA Longitudinal Simulation Capability for a Cruise Flight Condition," IEEE 1991 NAECON, May 24, 1991.

[5] Military Standard for Flying Qualities of Piloted Aircraft, MIL-STD 1797A, Wright-Patterson Air Force Base, Ohio, January 30, 1990.

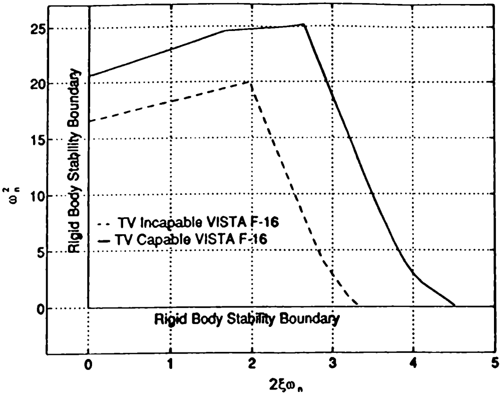


Figure 1 - VISTA F-16 Short Period Simulation Envelopes

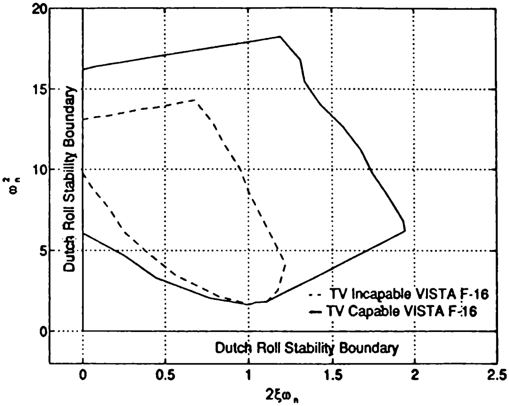


Figure 2 - VISTA F-16 Dutch Roll Simulation Envelopes

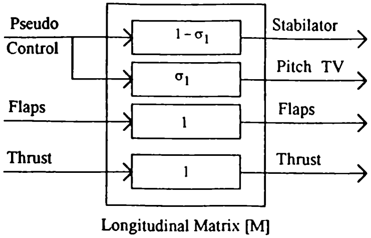


Figure 3 - Longitudinal Pseudo Control Mixing

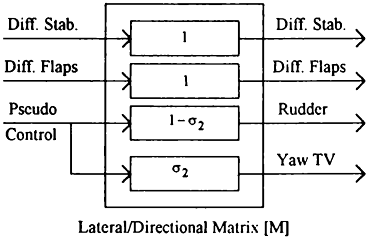


Figure 4 - Lateral/Directional Pseudo Control Mixing

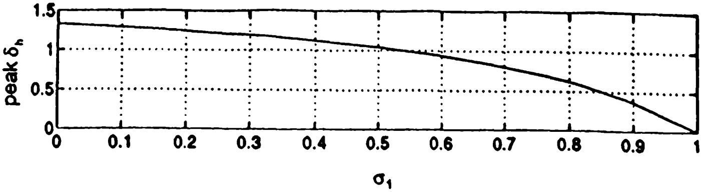


Figure 5 - Stabilator Peak Deflection Versus σ_1

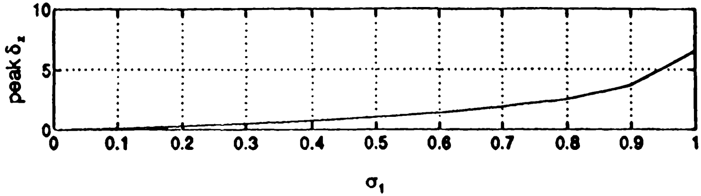


Figure 6 - Pitch Thrust Vectoring Peak Deflection Versus σ_1

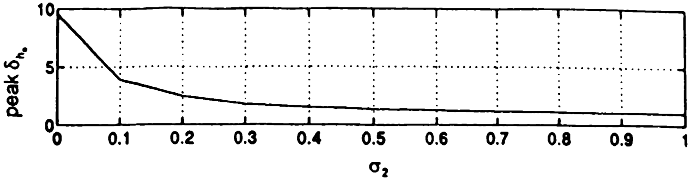


Figure 7 - Differential Stabilator Peak Deflection Versus σ_2

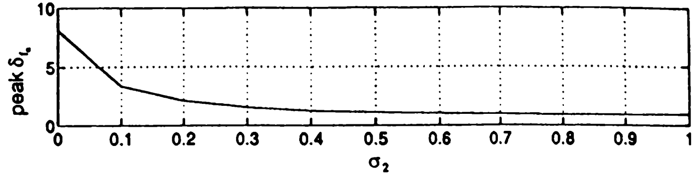


Figure 8 - Differential Flaps Peak Deflection Versus σ_2

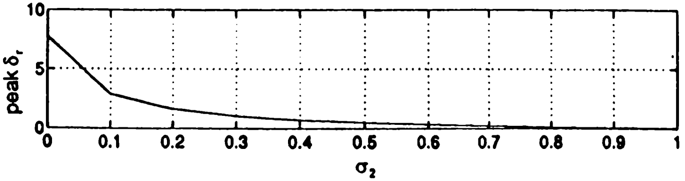


Figure 9 - Rudder Peak Deflection Versus σ_2

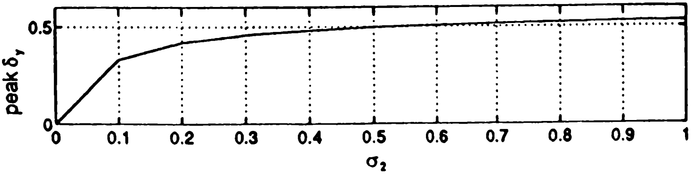


Figure 10 - Yaw Thrust Vectoring Peak Deflection Versus σ_2

AIAA-94-3422-CP

FLIGHT TEST DATA MODELLING AND COMPRESSION BY FUZZY EXTRACTED TOPOLOGY TECHNIQUE

Douglas D. Hensley
Boeing Commercial Airplane Group
Renton, Washington

Abstract

This paper describes a technique utilizing fuzzy set theory that the author has called Fuzzy Extracted Topology. This technique was used to create a reference model of an airplane's aerodynamic characteristics defined by flight test time history data. This reference model of the flight test data resulted in a substantial reduction in the amount of data required to represent the airplane aerodynamic characteristics.

Nomenclature

$C_{L_{SIM}}$	Simulator total lift coefficient
$C_{L_{FT}}$	Flight test total lift coefficient
C_X	Body x-axis lift coefficient
C_Z	Body z-axis lift coefficient
F_{XT}	Body x-axis thrust force, lb
F_{ZT}	Body z-axis thrust force, lb
n_X	Body x-axis load factor
n_Z	Body z-axis load factor
α	Airplane angle of attack, deg
$\bar{q}$	Dynamic pressure, lb/ft^2
S	Reference wing area, ft^2
W	Airplane gross weight, lb
M	Mach number
H	Altitude, ft
δ_E	Elevator deflection angle, deg
δ_S	Stabilizer position, deg
N	Number of independent parameters
M	Number of breakpoints for each independent parameter
L	Number of flight test data
μ_P	Proximity fuzzy set membership

Introduction

Updating the simulator aerodynamic math model is a highly iterative task requiring a significant commitment of engineering and computer resources. The motivation for developing this data modelling technique was to reduce the amount of resources required by the simulator update process. A major reason this update process is so burdensome is due to the manner in which the flight test time history data are currently utilized.

Current Flight Test Data Utilization

An airplane's aerodynamic characteristics can be described as a complex multi-dimensional surface, where each dimension is an independent parameter such as Mach, altitude, angle-of-attack, and elevator deflection. The simulator aerodynamic math model's function is to represent this complex multi-dimensional surface of airplane aerodynamic characteristics.

The simulator math model is implemented as a build-up of aerodynamic force components. For example, the total simulator aerodynamic lift, in airplane stability axes, can be expressed as[1]:

$$\begin{aligned} C_{L_{SIM}} = {} & C_{L_{BASIC}} + C_{L_{AEROELASTIC}} \\ & + C_{L_{STABILIZER}} + C_{L_{ELEVATOR}} \\ & + C_{L_{DYNAMIC}} + \text{SMALLER TERMS} \end{aligned} \tag{1}$$

Figure 1 demonstrates how two force component models of equation (1) aggregate to create a complex N-dimensional surface, where N is three for Figure 1.

Flight test time history data represent traces describing local regions of the N-dimensional surface of actual airplane aerodynamic characteristics (Figure 2). The values on the N-dimensional aerodynamic characteristics surface are calculated from the forces and moments required to satisfy inertial equilibrium for the kinematic state and mass characteristics determined from flight test measurements. For example, the total flight test data aerodynamic lift would be calculated in airplane stability axes as[1]:

$$C_{L_{FT}} = - C_Z \cos\alpha + C_X \sin\alpha \qquad (2)$$

Where:

$$C_X = \frac{n_X * W}{\bar{q} * S} - \frac{F_{XT}}{\bar{q} * S} \qquad (3)$$

$$C_Z = \frac{n_Z * W}{\bar{q} * S} - \frac{F_{ZT}}{\bar{q} * S} \qquad (4)$$

A flight test program will generate multiple sets of similar time history data for an array of maneuvers designed to isolate desired aerodynamic effects.

In a typical simulator model update, the flight test data are grouped by proximity of similar maneuvers, this grouping of flight test data traces creates a snapshot of specific local regions of the actual airplane characteristics surface. The aggregation of these data snapshots represent the total airplane aerodynamic characteristics surface defined by flight test data. During each iteration through the model update process, a comparison between the revised simulator model's surface and the actual airplane characteristics surface is provided by completely re-processing the flight test data groups. This method of defining the actual airplane characteristics surface by flight test data snapshots results in the repeated processing of nearly identical flight test data.

Flight Test Data Modelling

By aggregating the flight test data into a single N-dimensional surface model representing the actual airplane aerodynamic characteristics, this repeated processing of data snapshots can be avoided. The method used in this study to construct the N-dimensional surface model of the airplane flight test data utilized fuzzy set theory in a technique the author has called Fuzzy Extracted Topology (FET). Reference 2 is a recommended resource for understanding fuzzy set theory.

Implementation

A fuzzy membership set for data proximity was defined at every breakpoint, M, represented in the simulator component build-up models. At each flight test time history datum, the proximity fuzzy membership, μ_P, of the datum's independent parameter (e.g. Mach, altitude, angle-of-attack) was evaluated for each breakpoint. This represents the proximity of the datum to each breakpoint value. Figure 3 shows the calculation of μ_P for some representative flight test data and independent parameter breakpoints. The resulting proximity fuzzy set membership values, μ_P of Figure 3 are:

Angle-of-Attack	Mach	Altitude
$\alpha_1 = 0.25$	$M_1 = 0.00$	$H_1 = 0.15$
$\alpha_2 = 0.75$	$M_2 = 0.67$	$H_2 = 0.85$
$\alpha_3 = 0.00$	$M_3 = 0.33$	$H_3 = 0.00$

At every point in the NxM space containing the N-dimensional surface, a fuzzy logic rule was used to determine the contribution of each flight test datum to the definition of the surface at that point. This fuzzy rule was of the form:

IF μ_{P_1} AND μ_{P_2} AND. . . AND μ_{P_N} THEN
FET Value is FT Data Value

Where:
FT Data Value is the flight test based value of the surface being modelled

Using "product" as the composition operator definition for the fuzzy intersection, the evaluation of each rule became:

$$\text{Rule Weight} = \mu_P * \text{FT Data Value} \qquad (5)$$

Where:

$$\mu_P = \prod_{j=1}^{N} (\mu_P) = \mu_{P_1} * \mu_{P_2} * \ldots * \mu_{P_N} \qquad (6)$$

The contribution of each flight test datum defining the N-dimensional surface was accumulated into an implied rule base. This rule base was created by evaluating the fuzzy rule for each flight test datum at every point in the NxM space and aggregating the results using "sum" as the composition operator definition for the fuzzy union. From equation (5), the composed rule base could be represented as:

$$\text{Composed Rules} = \sum_{i=1}^{L} (\mu_P * \text{FT Data Value}) \qquad (7)$$

The de-fuzzified inferred value of the extracted topology was then calculated at every point in the NxM space as:

$$\text{FET Value} = \frac{\sum_{i=1}^{L} (\mu_P * \text{FT Data Value})}{\sum_{i=1}^{L} (\mu_P)} \qquad (8)$$

Thus, the sum-product inferencing of the implied rule base at each point in the NxM space reduced to a fuzzy weighted average model of the flight test data N-dimensional surface.

Results

An N-dimensional model of the Boeing 737 aerodynamic lift characteristics was created using this technique with flight test data in the region of Mach .72 to .78. The aerodynamic math model, Equation (1), for these data reduced to:

$$C_{L_{SIM}} = C_{L_{BASIC}} + C_{L_{AEROELASTIC}} + C_{L_{STABILIZER}} + C_{L_{ELEVATOR}} \qquad (9)$$

This resulted in a six-dimensional surface using independent parameters:

Independent Parameter Name	Number of Breakpoints
α	77
M	27
δ_E	12
δ_S	7
$\bar{q}$	7
H	4

The NxM space containing the six-dimensional surface consisted of 4,889,808 points.

The flight test data were composed of 150 individual conditions containing a total of 100,830 time history data samples with six parameter values at each data sample. The final 6-dimensional surface consisted of 8,846 points. This represents a decrease to 1/65th the on-line data storage capability required to model the flight test data defined actual airplane aerodynamic characteristics. The corresponding data processing time required for each iteration would be decreased by more than an order of magnitude.

Figure 4 shows a representative slice of the 6-dimensional surface compared to the snapshot group of data used to define the surface in the local region.

Conclusions

The current method of utilizing flight test time history data requires a significant commitment of resources to the task of updating the simulator aerodynamic math model. A reference model of the airplane aerodynamic characteristics defined by the flight test data can be created using the Fuzzy Extracted Topology technique. This reference model resulted in a substantial reduction in the amount of data required to represent the flight test data airplane aerodynamic characteristics and an order of magnitude decrease in the data processing time required for each iteration of the simulator update process.

Acknowledgements

Very special thanks to Terry Schulze of the Boeing Commercial Airplane Group Aerodynamics Engineering for his enthusiastic support which made this work possible.

Special thanks also to Brian Bocksch and Brett Churchill for their support and encouragement.

References

1. Neville K.W. and Stephens A.T., Flight Update of Aerodynamic Math Model. In AIAA Flight Simulation Technologies Conference (Monterey, California 1993).

2. "Fuzzy Set Theory and Its Applications", Zimmerman H.-J., 1991, Kluwer Academic Publishers.

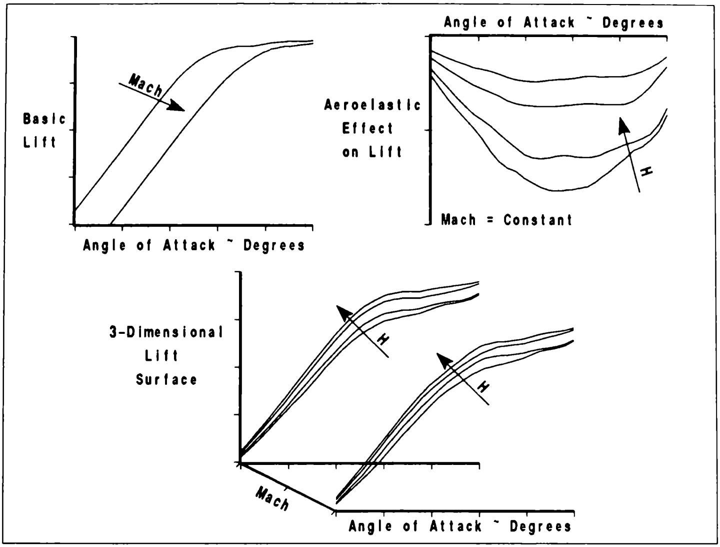


FIGURE 1. Complex Surface by Force Component Build-up

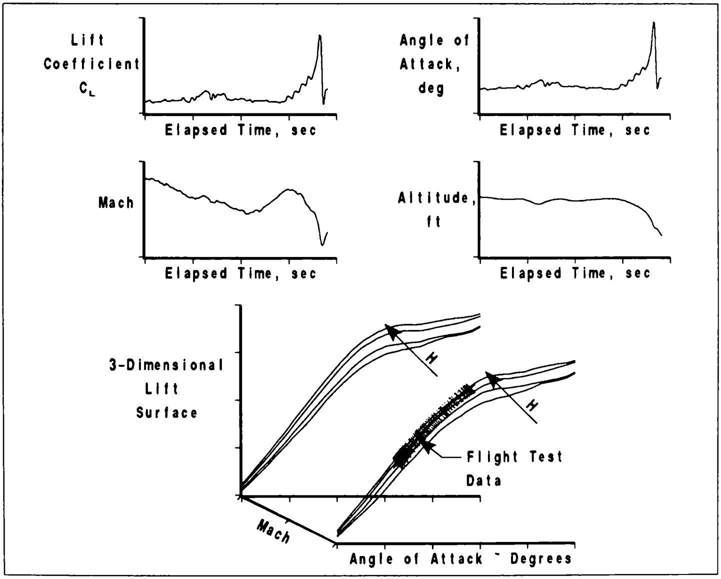


FIGURE 2. Flight Test Data Trace on Complex Surface

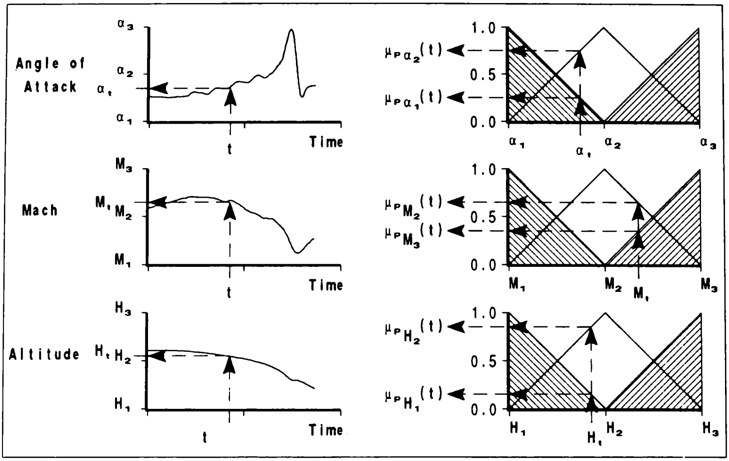


FIGURE 3. Proximity Fuzzy Membership Calculation

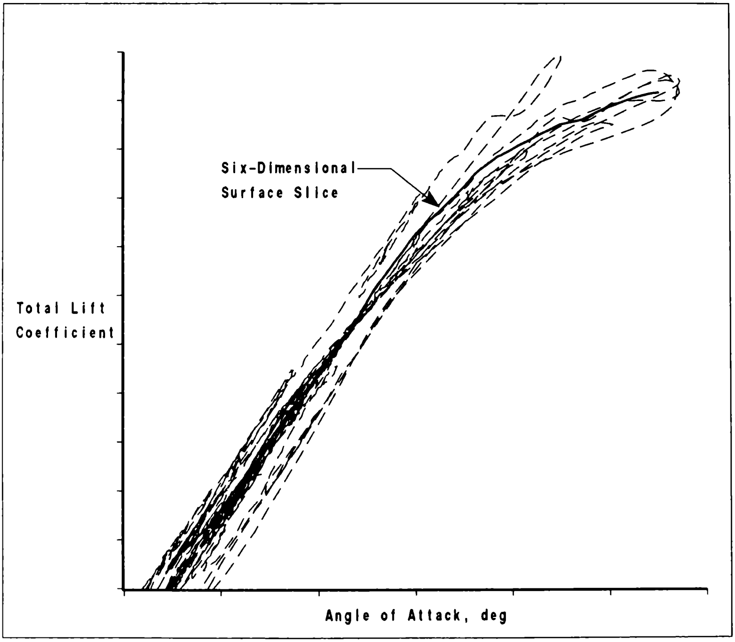


FIGURE 4. Representative Slice of Six-Dimensional Surface

AIAA-94-3423-CP

OPERATIONAL PROTOTYPE FOR AN INSTRUCTOR/OPERATOR STATION

Dick Fulton and Ankur Hajare
Enhanced Technologies
Houston, Texas

Tom Diegelman
NASA Johnson Space Center
Houston, Texas

Dave Webster
CAE-Link Corporation
Houston, Texas

Abstract

This paper describes the implementation process for technology insertion into a real-time, human-in-the-loop, flight simulator of the Space Shuttle used for astronaut training. The Instructor/Operator Station (IOS) is a twelve year old, highly tailored subsystem that was not designed to easily accommodate changes in hardware and software technology. Since the Shuttle program is anticipated to run another 15 years, the objectives of the project were to identify and evaluate commercial off-the-shelf (COTS) hardware and software that meet defined requirements for the upgrade of the IOS in the training facility. The rationale for conducting the prototype was to find the best possible way to upgrade the IOS and minimize the life cycle costs for continuing operations. The paper illustrates the prototype architecture that was successfully used to establish the confidence of NASA management in the concept and to refine the technical requirements for the IOS upgrade. The paper also discusses the lessons learned in implementing an operational prototype in a complex real-time simulator as well as the plans for the future of the IOS.

Introduction

The principal reasons we were successful in developing the IOS prototype is that the challenge associated with the concept of injecting new technology into the Shuttle Mission Training Facility (SMTF) was accepted as an innovative task that would become a cornerstone in a very long project.

As an agency, NASA is still quite "young" and being less than 30 years old, the JSC facilities are likewise "young." These facilities were built when the display technology was just beginning to mature. As a trainer built 15 years ago, the SMTF exhibits custom designed, specialized equipment that today would likely be purchased and installed as COTS.

Our facilities are tailored to concepts of what is both needed and feasible. Vertical integration of the facility is still evolving. This creates a large, complex set of equipment that meets the objectives of only a single program. We are faced, for the first time, with the reality of working two long term space programs concurrently. As the Space Station Freedom program comes on-line, we still must support the Shuttle program.

We need to seek out facility upgrade concepts that reflect common hardware, software, and architecture across multiple programs. The training facilities being built for the Freedom program will be mostly state-of-practice COTS products. It is all too obvious that the same COTS products could also be used in the upgrade of the SMTF and make substantial reductions in the cost of ownership for both facilities.

The IOS prototype team was required to identify the facility issues associated with technology insertion and accommodate the ongoing needs of the instructors, operators, and maintenance personnel during stand-alone and integrated training.

During integrated training, the Network Simulation System (NSS) provides a realistic representation of the real world Shuttle communications network. Details, such as loss-of-signal occlusion, are modeled in this simulator. The NSS is sufficiently realistic, that during integrated simulations (MCC in the loop) the ground controllers in the MCC can not tell if the Shuttle is actually flying or being simulated in the SMTF.

Instructors are required to enter reconfiguration commands in a short period of time as the simulated Shuttle communications move from one ground station to the next. As we prepared plans and requirements for the upgrade to the SMTF, the operators wanted a more efficient method of entering NSS reconfiguration commands. To really make this work, the operators proposed using workstations to provide this improved functionality.

It did not make sense to us to put in a workstation for an isolated user. We considered it essential to develop a prototype and install it in the training environment to assess the user evaluation of a new IOS subsystem. This would give us broader exposure to the new technology and provide a much more cost effective way to gather needed requirement data.

Background

Training in the SMTF is designed around the concept of remote instruction. The instructor is not in the seat next to the trainee with direct override capability. This override capability must be simulated.

The SMTF architecture uses a central host computer system and a relatively "dumb" man-machine interface for instructors and operators. The SMTF host runs on a 40 millisecond time frame and the frames are becoming saturated. This is evident from soft transgressions. Without a complete restructure of the simulation design, it is difficult to reconfigure from flight to flight and add any nuances. This points to a need to off-load the host.

The IOS is the focal point for external interfaces in a real-time simulation. The instructors must have interactive access to the data observed in the crew station, and the operators must have the resources to control the real-time simulation loads.

All functional interfaces such as visual scenes, voice communications, flight aural cues, on-board computers, and others, must be available in the IOS.

Given the state-of-the-art in the 1970's, the requirement for even the most basic IOS required development of in-house software and hardware products. We needed to understand the existing simulation interfaces in detail. We also needed to understand the COTS products to effectively design the interface between the COTS products and existing systems.

It is important to maintain current capability while designing the in-roads of new requirements.

Little of the SMTF environment is static. Flight reconfiguration and facility modifications are needed for each flight. Many of these modifications are payload dependent.

We create a software baseline to target subsystem replacements. Baseline one is used to sustain training, and baseline two is used to implement development changes. Changes made to baseline two are handed over to operations when development is complete. In the SMTF, baselines are created at six month intervals. If we miss a baseline drop, we face a possible six month slip in our software development schedule.

We need to understand and be aware of other ongoing changes in the facility. We are convinced that this is critical to the success of any prototype. Even things we believe are cast in concrete, can be changed. We will discuss one such surprise later.

Upgrades to training capability are dependent on whether or not the existing equipment can support the change.

The IOS equipment is marginally supportable. Hardware parts are mostly available, but the systems are no longer sold as stock items. The end of the life cycle for the IOS hardware is near. Because of the custom tailored software coupling to the hardware, the software is "doomed" as well. Porting of this software is not possible. The data display systems in the IOS have been in use in excess of fifteen years. We developed the IOS page compiler in-house. It is written in assembly language on a 36 bit computer. This is a hardware and software dinosaur, and it is also not portable. This hardware and software is like a piece of flypaper, once you get hold of it very difficult to financially justify getting rid of it. We documented this in the Level A Functional Requirements document (NASA Johnson Space Center, 1991).

Page development requires simulator time. The page code is written off-line but simulator time is required to compile and test the page code and checkout the format of the page. If a page requires a change, the simulation must be brought down, the change made off-line, and then the simulation load must be rebuilt to include the page modifications. Changing pages in a training load is expensive due to the load building overhead. We rarely make small changes due to the large cost of simulator usage. Instructors are forced to work with page code that has technical problems. What should be an inexpensive task, is not, using the simulation host is driving costs up.

When the SMTF was built, we built a custom interface between the IOS and the host computer because high interrupt rates from the IOS were causing the simulation to terminate. The interrupts from the IOS had to be metered to a rate the host computer could support.

Approach

We wanted the new IOS to be a compatible (a plug-in component) with the both the existing and a potential COTS replacement simulation host.

We wanted to off-load the data conversion and presentation work from the simulation host computer to the IOS. This workload is very unpredictable and it makes sense to move it outside of the timing-sensitive simulation host. We decided that the MCC design, that included a Network Data Driver (NDD) with suitable "enhancement", would satisfy the IOS upgrade requirements.

The Configuration

The NDD serves as the networking "traffic cop". It intercepts, analyzes, and validates data requests from the instructor position.

The host computer data, returning to the instructor position, is converted to required format by the NDD. There are two Local Area Networks (LANs) that are controlled and monitored by the NDD. The real-time LAN distributes data to the instructor positions. The general purpose LAN is used for full dialog between the NDD and the instructor position.

The hardware interface between the NDD and the simulation host is COTS, used by many vendors.

The software interface is through the symbol dictionary The symbol dictionary provides the memory address translation between the requested data term and the virtual memory of the simulation host. This feature provides us the ability to build displays outside of the simulation host, makes the IOS upgrade concept possible. Regardless of the re-host approach, there will always be some form of change of the data output by the simulation host. Any changes to the NDD is minimal and localized to the address resolution software or the data conversion routines.

The Team

We had a good combination of talent and dedication.

The prototype team consisted of five engineers. The Project System Engineer was responsible for budget, schedule, and customer interface. One System Engineer was responsible for the engineering of the technical aspect of the task and specifically the NDD.

One Engineer was assigned to the workstation. He was responsible for the code and pages running on the workstation. This engineer was also responsible for all the software administrative functions. One hardware engineer was assigned to the hardware interface to the simulation host. Another engineer was responsible for "hooks" into the simulation code for data gathering

The Users

Because few upgrades in technology have been implemented in this time frame, the impact of technology insertion is very significant to the users.

The IOS upgrade is a unique departure from the past. It is mandatory to implement changes in such a way that the user can be brought "on board" in small steps.

We wanted the users to help create the design. This will make the end product more acceptable. We asked the users to play a critical role in the requirements definition and design phases. Users were brought in weekly to work with the prototype. Great care and attention were given to the user inputs, making the user an integral part of the process.

We initially emphasized replicating the capabilities of the IOS as the instructors see them today. The users identified a core set of displays needed to perform a majority of their routine activities. This core set was developed in a "look-a-like" fashion.

The users are given small prescribed doses of change to keep familiarity alive and not perturb every day astronaut training.

We introduced new twists about every three months. For example, video that requires its own monitor was routed through a "window" to the new display hardware. This was not a particularly large change, because the scenes had not been modified, but great care was given to make sure that the new IOS capability would be compatible with the visual upgrade going on in parallel.

Results

Preparation

Our first, and perhaps most difficult, task was to obtain management approval for the prototype. This process involves preparing a design package that includes the following significant parts:

- Required floor plan modifications
- Electric power needs
- Temperature control, chilled water and air
- Data path identification
- Security and equipment isolation

We worked on the approval process for two years. During this time we made one major change to the design package. This change added a second training base where the prototype would be installed.

Also during the approval process, we proceeded to establish the initial prototype in the Link lab. We built a stripped down real-time load and took crude timing measurements.

Next, we had to convince the SMTF sustaining contractor that what we were adding to the simulator host computers was a small and manageable risk. The SMTF has two temperamental processes.

- All tasks must be completed within 40 ms
- I/O Channel use has little to spare

We worked with the operations and sustaining engineering people to gain their confidence in our plan.

Installation

We go through a process that assures we cover all important aspects of change to the SMTF. In some cases we need the specific technical details for change from other organizations, and in other cases we have to supply the technical detail ourselves. This process may be different for you because of the organization of your support services.

Installation of the prototype was done in three stages. First, we needed a development facility where initial work could be done without the risks of interrupting astronaut training. The first prototype was installed in the development contractors facility. We were able to build a realistic capability for demonstration, but we could not tie into the real time data available from the simulation host computers. We used this capability to increase user, management, and sustaining contractor confidence in the prototype. This installation also provided an opportunity for the development contractor to learn a different COTS programming language.

Second, we installed the prototype in a training base that doubles as a development facility. This installation required close coordination with other SMTF upgrade activities that were being done at the same time. This training base is being upgraded to full training capability. At one point, we were planning a demonstration that conflicted with dismantling activities on that training base.

This second installation provided the first opportunity to bind the prototype to the real-time data flow without the risk of interrupting training. At this point, we could and did demonstrate the enhanced capabilities proposed for the IOS upgrade.

Third, we installed the prototype in an active training base. This installation gives the instructors an opportunity use the prototype during a real training session. It also provides a way to get user feedback. This feedback closes the loop on integrating the user in the IOS upgrade development process.

The third installation came with several caveats. We were not to cause the simulation to terminate abnormally. (Murphy's Law "If it can happen, it will happen" got to us - we did it!). We were not to run the prototype during an integrated simulation involving the MCC and there by risk a much more costly simulation termination (Murphy got to us again!). We suffer from the "new guy on the block" syndrome. We get blamed for any unusual problem in the simulator.

We spent several weeks trying to discover what went wrong that caused the simulation to terminate. Early in the investigation, we discovered that the data buffers were corrupted. Thus, we found out that some other upgrade task broke the concrete!

We set our buffer switching algorithm based on the 25 hertz frame dispatching rate. Due to another problem in the simulator, the first frame at simulator initialization was skipped. This caused our buffer switching algorithm to use the same buffer for both input and output. We had to wait for the next six month baseline delivery to implement a correction for this buffer switching problem.

Conclusions

We determined that the MCC network data driver and workstation configuration can be used in the SMTF. The generic configuration is shown in figure 1.

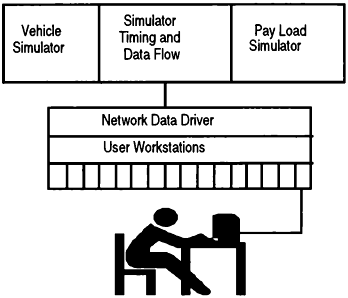


Figure 1
Generic IOS Configuration

You should note that the implementation of this configuration is highly dependent on the geography of the training facility. In the closest of spaces, you may be able to package the simulator host computer, the network data driver and user workstations in the same 19 inch rack. The operations and maintenance people will love it.

Metrics

We accomplished the goal of identifying requirements for the next generation IOS subsystem.

- The total simulation host computer load consists of about 600,000 lines of code. About 200,000 lines of code are executed every 40 milliseconds. The prototype added 5,000 lines of code to the 40 millisecond work load.
- The data path between the simulation host computer and the network data driver must be able to sustain a synchronous data path capable of transferring 1 megabytes per second.
- The network data driver must execute 2,500 lines of code every 40 milliseconds.
- The network data driver must be able to sustain a synchronous data path capable of transferring 1 megabyte per second.
- The network data driver must be able to handle a peak asynchronous data rate of 25 messages per second with a maximum of 8,000 bytes per message.
- The IOS must be capable of continuously logging data at rate of 8,000 bytes per second for each of a maximum of twenty five users. The NDD must be able to log time stamped events up to a maximum of 125 events per user per second.
- The IOS must be able to execute 12,000 lines of source code per user per second.
- The IOS must be able to display in color a maximum of 12 windows (2,000 single precision data terms) including one video scene per user per second. All data shown on user displays must be time homogeneous, this means text and graphics.

The network data driver must be benchmarked at 40 "SPECMARKS". Each user must also have processing capability bench marked at a bare minimum of 25 "SPECMARKS". We recognize that the user processing needs will increase with new training requirements and with upgrades in COTS software products. We believe it is desirable to find a computer vendor that has a linearly scaleable product. Fortunately,, there are several vendors that offer single board computers that can be upgraded with new technology at the board level.

Lessons Learned

We had one situation we could have handled better. At the time we were seeking approval for the IOS prototype, we were also participating in another prototype with a more limited scope but addressing the same issue - "Improving the User Interface to the SMTF". The other prototype was targeted specifically at display design using COTS graphical products. This prototype had user support, money, equipment, and a small but talented staff.

A significant difference between the two prototypes was the use of real time data to drive the displays. The other prototype was using mock-up displays that did not react to the dynamics of any real time data flow. We attempted to merge the two prototypes into a single effort, but we were unsuccessful. The other prototype was dissolved, the staff was reassigned, the money was withdrawn, and the equipment was reallocated to other needs. What remained was the user support and the documentation of their accomplishments to date.

At issue was the type of funding between the two prototypes. Funding for the IOS prototype was development, and funding for the other prototype was research. Also at issue was our need to scope the amount of effort we needed to build user displays. In retrospect, we should have scoped the full effort for building user displays and taken advantage of anything the other prototype could have provided. By drawing attention to the similarities and differences of the two prototypes, we lost the support and possible contributions the other prototype could have provided.

We sent a very clear message to the user and maintenance community. Even in this completely "in-house" software environment like in the SMTF, COTS products could be introduced with minimal difficulty and yield superior results. This is one of the most significant effects the IOS prototype.

The IOS prototype was developed in a very open environment. Periodic demonstrations, working group meetings with the users, and hands-on user involvement were intricate parts of the IOS development. The demonstrations were scheduled on the basis of capability deliveries. Each delivery was reviewed by the users, written comments were taken and the users were able to mold the product through the development cycle. This development method keeps the user community involved.

Recommendations

We did some things correctly and we want to share these with you.

Prepare an implementation plan that shows how the prototype will evolve. There are three important points that attract attention. First, be specific about the expected results and deliverables of the prototype. Second, the plan should include several demonstration sessions so management will recognize significant progress when reported. Third, show that the prototype has a definite development termination date, even if it is planned to be left in an operational state.

Select the prototype team carefully. The lead team members must be convinced the effort will produce worthy results, that the approach is technically sound and attainable goals set within the plan schedule. There is a natural conflict of interest between the selection of prototype personnel and the determination of the prototype schedule. In order to shorten the prototype schedule and produce meaningful results, you will select your best performers to work on the prototype. On the other hand when problems arise in profit centers, these same people will be pulled away from the prototype to solve these problems. A lingering prototype will eventually be determined unsuccessful.

A successful prototype does not come without commitment.

Involve the technical lead personnel in the preparation of the prototype implementation plan. Listen to them; after all, their careers and credibility are also at stake, and should be. People who are committed to the successful completion of the prototype are the only real candidates. Risk avoiders should be avoided.

Avoid committing to a comprehensive set of documentation requirements. The prototype plan should be synchronized with the development schedule. Deliverables from the prototype should meet the needs to document the requirements of the development project. Let the development project bear the cost and schedule impact for comprehensive documentation. Documentation needs will refine themselves as the prototype effort evolves.

Be prepared to defend the prototype at every status meeting. Your efforts can be attacked from the most unsuspecting corners of your environment at any time. There is an ancient saying "Under the tree with the ripest apples, lie the biggest clubs." If the prototype is not attacked, it is probably seen as a failure in the making.

The Future

The vision for the future of the SMTF complex with respect to the IOS has been historically difficult to formulate. However, recently NASA has made agency wide and center wide attempts to establish guidelines, much analogous to the "why and how to prototype" guidelines the authors established during the IOS prototype effort. There is clearly a charter to infuse technology, but in a manner consistent with information system technology in NASA facilities.

"The NASA of the future will not rest on its laurels, nor will it shrink from attempting the dangerous and difficult. Its torch will be kept burning by its most precious asset -- the young men and women drawn to NASA by its bold and shinning vision." (ref 2, pg 3)

The priorities defined are to "restore, maintain and construct, in that order" information systems.

"Besides the physical plant, NASA must also maintain, upgrade, and replace equipment, everything from computers and rocket engine test stands, to research and mission aircraft and life sciences research equipment." (ref 2, pg 23)

In the interest of attaining the agency wide goal of a sustained 15% Shuttle Program cost reduction by 1996.

"NASA is studying and implementing various improvements in the Shuttle program to achieve a 15-percent reduction in operations cost by 1996 and to improve the avionics, propulsion, and engine components." (ref 2, pg 32)

Clearly innovation is required. This translates into improved facility management philosophy, especially with respect to information processing centers

"No technology arena is changing faster than those involving communications and information processing." (ref 2 pg 34,37).

In addition to the agency initiatives, the JSC objectives renew the commitment to "build on the existing systems" with a focus on long term, "30 year focus".

"Programs that involve people actually living and working in space will demand entirely new approaches to how we think about programs, how we develop them, and how we operate them." (ref 3 pg 7).

To tie dissimilar facilities and systems together is most easily accomplished under the auspices of common procurements for equipment, allowing divergent applications and operation environments to function with common hardware and software, most of which are commercial-off-the-shelf (COTS).

Against this vision, the "futures" of the IOS developed. A brief discussion of the major initiatives identified and considered to be candidates for follow-on system engineering studies are discussed.

With the inclusion of high powered workstations into the IOS platform, the ability to utilize LAN connectivity introduces the separability of the individual workstation to perform dedicated tasks. The merging of the single system trainer functionality (SST), currently a host / dumb terminal based system, into the full fidelity SMTF is seen as very attractive. The necessary models to create the SST environment would require dedicated MIPS, models, sequencers, user interface, and a rather low-fidelity functional flight software model as a "driver" for the entire system . In place of the "hard" configuration of switches and panels, the workstation's multiple graphics heads form the "panels".

The entire system is then linked together by the predetermined sequencer (i.e., the requirements for what system is being trained) and is executed stand-alone from the full SMTF base. If multiple students required multiple stations, multiple workstations would be networked together. Again, this would be a predefined sequencer that "built" the environment. Upon completion of the session, the workstations are reconnected to the LAN (figure 2).

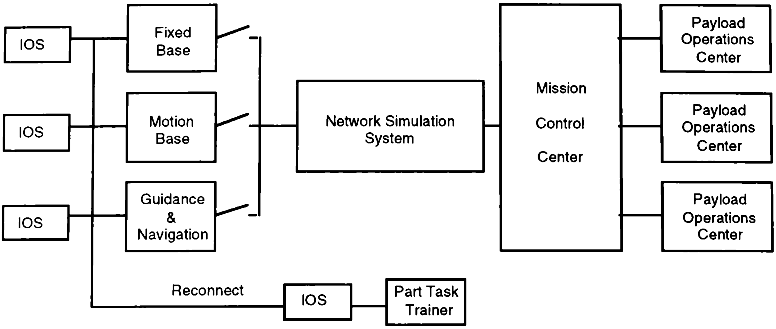


Figure 2
Facility Level Configuration

References

NASA Johnson Space Center, 1991, Level A Functional Requirements for the Shuttle Mission Training Facility Equipment Replacement or Upgrade Step 4, (Draft), Simulation Systems Branch, Houston, TX.

National Aeronautics and Space Administration, 1992, Vision 21 The NASA Strategic Plan, Houston, TX..

NASA Johnson Space Center, 1992, The JSC Strategy, Houston, TX.

AIAA-94-3424-CP

A FACILITY FOR SIMULATING SPACE STATION ON-ORBIT PROCEDURES

Ankur R. Hajare, Daniel T. Wick, and Nagy M. Shehad
Enhanced Technologies
Houston, Texas

Abstract

NASA plans to construct the Space Station *Freedom* (SSF) in one of the most hazardous environments known to humankind - space. It is of the utmost importance that the procedures to assemble and operate the SSF in orbit are both safe and effective. This paper describes a facility designed to train the *integration* of the telerobotic systems and assembly using a real-world robotic arm grappling space hardware in a simulated micro-gravity environment.

Introduction

NASA plans to construct the Space Station *Freedom* (SSF) in one of the most hazardous environments known to humankind - space. It is of utmost importance that telerobotic systems and procedures to assemble and maintain the SSF are both safe and effective. In addition, it is equally important that the mechanical attachment devices of the SSF function as planned. To ensure that the systems and procedures will meet all safety requirements, NASA is developing the Space Systems Automated Integration and Assembly Facility (SSAIAF). This facility will test the *integration* of flight systems, the telerobotic systems, and humans that will be involved in the assembly and maintenance of the SSF. Both mechanical and dynamic performance of these elements in a micro-gravity environment will be tested using real-time simulations.

Traditionally, the approach to building such hardware-in-the-loop, human-in-the-loop simulations has involved development of unique software to support each scenario. This scenario-specific approach results in continuous software development and maintenance, which consumes large portions of a project's budget and can adversely affect a project's schedule. A proposed SSAIAF software architecture, Simulation and Control Environment (SCE), will support the development of generic software components, called modules, that can be reused in numerous simulation scenarios. This approach will reduce the requirements for new simulation development.

This paper presents a brief overview of the SSAIAF. This is followed by a more in-depth description of the SCE proposed to be used by SSAIAF in developing the real-time simulations. The paper concludes with a description of how the SSAIAF can support the development of telerobotic procedures and space systems.

SSAIAF Systems

SSAIAF systems will provide the resources for all testing and development activities undertaken in the facility. The SSAIAF systems are shown graphically in Figure 1.

Computing Resources

Computing systems in this environment include hardware with sufficient processors, memory and I/O capacity to support two simultaneous tests on SSAIAF systems. The currently envisioned system will include three host computers, peripherals, and computer control consoles.

Manipulator Development Facility

The Manipulator Development Facility (MDF) is a stand-alone system that provides a realistic test and training environment for astronauts who will use the Shuttle arm. The MDF permits astronauts or test engineers to lift structural components out of a mockup of the Space Shuttle Orbiter cargo bay and position them for deployment or erection.

Mobile Remote Manipulator Development Facility

The Mobile Remote Manipulator Development Facility (MRMDF) is a stand-alone system that provides a realistic test and training environment for astronauts who will use the Space Station arm. The MRMDF permits astronauts or test engineers to grapple and position a station payload, a platform, and another object for deployment or erection on the Space Station.

Air -Bearing Floor

The Air-Bearing Floor (ABF) is a large, smooth surface area (70' by 90') upon which pallets can float on a cushion of air. Each pallet can carry over a thousand pounds of weight. The cushion of air permits the pallets to move in two dimensions as if there were no friction. Thus, the air-bearing floor simulates weightlessness in two dimensions.

Six-Degree-of-Freedom Dynamic Testing System

The Six-Degree-of-Freedom Dynamic Testing System (SDTS) is a computer-controlled hydraulic table (Stewart Platform) that can move in six planes of motion at one time (right/left, back/forward, up/down, pitch, yaw, and roll). The SDTS can lift large objects weighing up to 10,000 lbs. and move them to a position, as directed, within a tolerance of 0.1 inch. The orientation of the SDTS is such that it usually moves objects vertically.

Simulation and Control Environment

The Simulation and Control Environment (SCE) is a software system that integrates all elements required for the development of SSAIAF real-time simulations in a comprehensive and cost effective application generation environment. The modular approach allows the reusability of software developed for previous simulations. The SSAIAF simulations developed using the SCE will execute on the test computing systems. SCE development activities utilizing a user-friendly, icon-driven user interface will be accomplished on development/maintenance workstations. The SCE architecture and usage is further discussed below.

The Simulation and Control Environment

SCE Architecture

The proposed software architecture for SCE will include both Commercial-Off-the-Shelf (COTS) and custom components. SCE uses a layered architecture (Figure 2) that supports modular and reusable software modules as well as application generation techniques that provide fast development and ease of change. The SCE has been designed in a modular architecture for three reasons. First, this modular approach supports the maintenance of the entire system. This permits system maintainers and developers to revise and update any particular module without affecting the entire SCE. This modular design is also POSIX compliant so that it can be easily migrated to other platforms as hardware continues to evolve.

The second reason for this architecture is to support an automated approach to the development of simulations. The SCE is designed to maximize the reuse of previous development efforts. To this end, a developer can select a highly developed software module or portions of previous simulations and incorporate those in a new effort. For example, a developer could select arm models from one simulation and propulsion models from another and combine those in a new simulation. Thus, a developer has the ability to reuse software at any level that is most appropriate for the development effort being undertaken.

The third reason for this architecture is to facilitate simulation generation by automating many attributes traditionally determined by the simulation designer. For example, a simulation executive/scheduler within the SCE will automatically allocate software modules (processes) to one or more CPUs communicating through shared memory, in order to maximize the efficiency of the simulation. In the traditional approach these decisions would be made by a designer in an arbitrary way and evaluated in a trial-and-error manner.

Major SCE Elements

The SCE is comprised of four major elements (Figure 3). First, the math models are implemented in the form of modules and can be considered as building blocks for the simulation developer. The modules themselves are built using lower level primitives. The second element is an icon-based user interface that allows the simulation developer to define a particular simulation using a flow diagram-like notation that expresses the data-dependencies between the various modules, the simulation rates, delays, etc. The data representing a particular simulation are then stored as a set of Simulation Definition File(s). The third element is an application generator that translates the data created by the simulation definition element into another set of files called the Simulation Data File(s) and Executive Input File(s). The Executive Input File(s), which are generated following test computer dependent directives, are then shipped to the test environment, in addition to the simulation initialization data. The fourth element of the SCE, which is resident on the test computer complex, uses the Executive Input File(s) to setup the execution environment at run time and actually schedules and controls the simulation and automatically allocates parallel processes (when available) to multiple CPUs.

Simulation Life Cycle Using SCE

The use of the SCE in the simulation life cycle is displayed in Figure 4. The life cycle begins with the generation of simulation requirements. If existing SCE capabilities cannot support these requirements, SCE or Module developers will develop additional capabilities, e.g. a robot model, to support the simulation requirements.

Once the necessary capabilities are available, a simulation is defined and developed. This might include combining a Space Station Freedom model, a robot model, and ephemeris modules (gravity, solar, etc.) into a simulation. The simulation is generated using predeveloped modules and tested. Once the simulation is verified, the end user will run the simulation and evaluate the results.

Developing On-Orbit Procedures

Rapid Procedures Development

The SCE will support rapid prototyping of procedures by permitting the user to select modules ranging from simple to complex, and to combine the modules in a simulation to investigate the action of these systems in space. The system will contain modules for the Space Station systems and environment (gravity and solar), and it will automatically combine these complex modules into a testable simulation. These will be displayed in the trainer's workstation screen. The trainer will then run a simulation using various scenarios.

Procedures Test and Verification

The SSAIAF will be designed to move flight-like articles as they would move in space to provide form-and-fit verification testing over a range of possible conditions. The SCE will support the evaluation of the complex dynamics and forces that cannot be tested elsewhere. The procedures using actual space systems can be tested utilizing SSAIAF test systems. Repeated and long term testing is possible, as well as off-nominal operations (e.g., collisions).

Astronaut Training

Upon procedure verification, the SCE will support training of astronaut-robot teams to ensure that their tasks can be accomplished in a safe and efficient manner. Experience has shown that real-time simulation cannot provide all the hands-on training required to successfully complete mission training. Astronauts will be trained in the SSAIAF using either real space systems or volumetrically identical mock-ups. In addition, on the Space Station humans will have to work in conjunction with a robotic systems. Thus, actual or simulated testing of human-robot teams must be performed. The SSAIAF will provide as close to a realistic training environment for astronaut-robot teams as possible on earth.

Problem Resolution

Experience in space operations has shown that anomalous situations occur which require real-time resolution (e.g., utilizing a robot in an untested scenario). The SCE could support rapid prototyping of possible solutions and testing of untested robotic operations. Interfaces with Johnson Space Center operations could include data and voice links with Shuttle and Space Station astronauts as they try to resolve on-orbit problems.

A Repository of Simulation Objects

Finally, the SCE will serve as a repository of space systems models that can support the training required by future space systems. Future space systems will be designed and tested. The SCE will support the archiving of simulation modules and the results of prior testing for the benefit of future training program developers.

Conclusion

Space is one of the most hazardous environments in which humans must work. The difficulty of assembling, maintaining, and repairing systems has led NASA to develop a facility where the integration of procedures and systems can be tested simultaneously. Testing space systems prior to launch requires a simulated environment. As stated above, the traditional approach to building, such as human-in-the-loop simulations, has involved development of unique software to support each scenario. However, in the SSAIAF the SCE approach represents a departure from tradition. The modular design maximizes the reuse of previously developed simulations. An entire simulation need no longer be developed from "scratch." Instead, developers can combine components of previous efforts and concentrate their efforts on the unique elements of the new simulation. In addition, the modular design of the SCE supports its own maintenance and permits migration to new platforms as hardware continues to evolve. The result of this approach is an adaptable software environment that supports the rapid development and test of space systems and the procedures they require in a cost-effective manner.

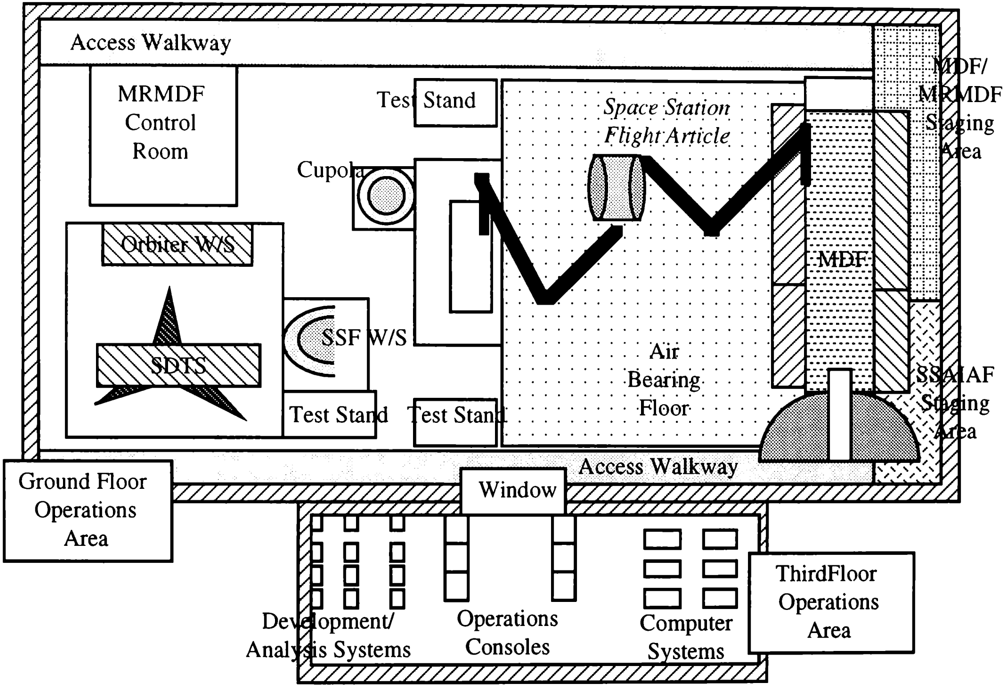


Figure 1. SSAIAF Systems

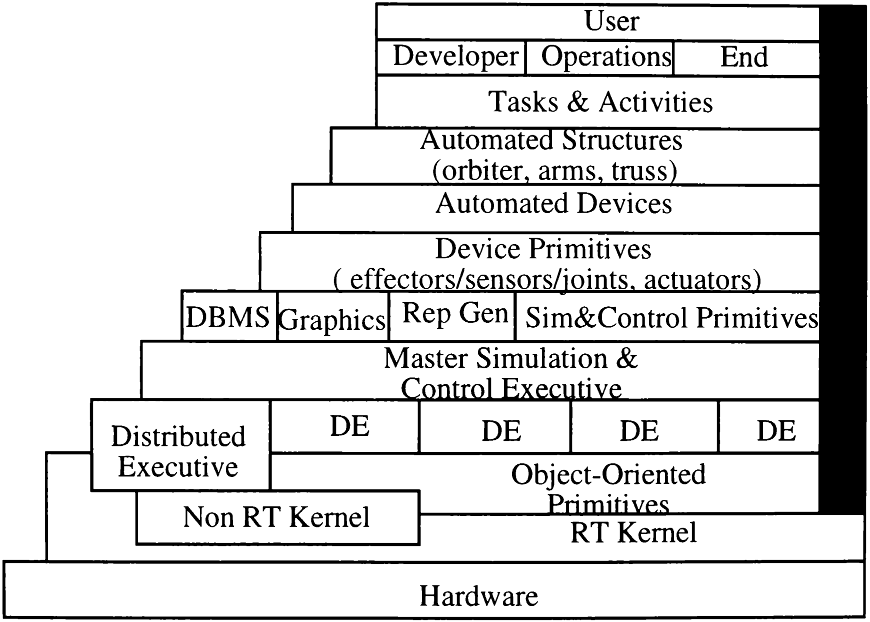


Figure 2. SCE Modular Design

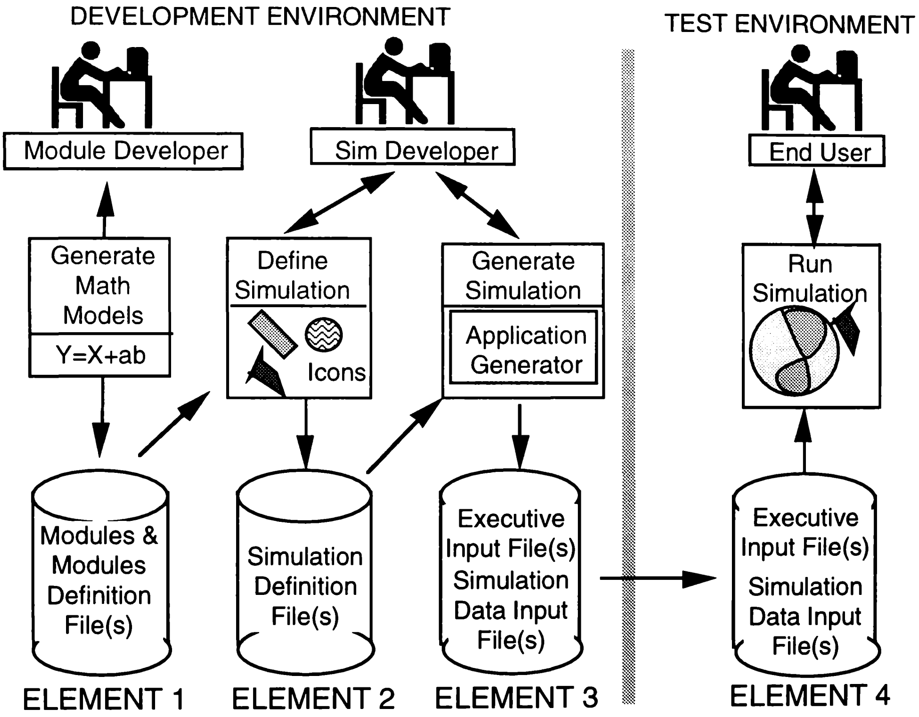


Figure 3. Major SCE Elements

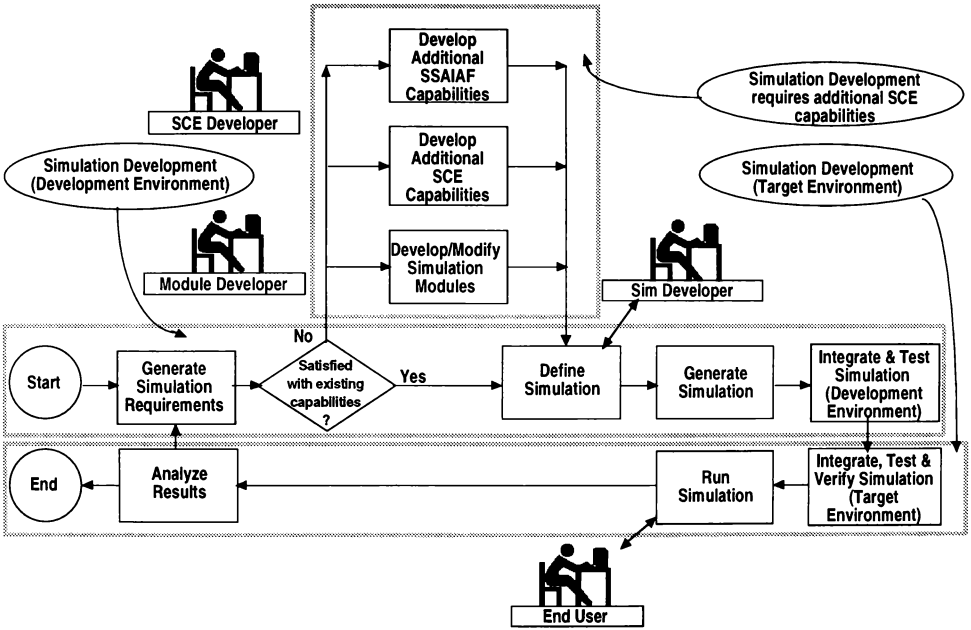

Develop Additional SSAIAF Capabilities
Develop Additional SCE Capabilities
Develop/Modify Simulation Modules
SCE Developer
Simulation Development requires additional SCE capabilities
Simulation Development (Development Environment)
Simulation Development (Target Environment)
Module Developer
Sim Developer
No
Start
Generate Simulation Requirements
Satisfied with existing capabilities ?
Yes
Define Simulation
Generate Simulation
Integrate & Test Simulation (Development Environment)
End
Analyze Results
Run Simulation
Integrate, Test & Verify Simulation (Target Environment)
End User

Figure 4. Simulation Life Cycle Using the SCE

AIAA-94-3425-CP

Study on the Network Load in Distributed Interactive Simulation

Kuo-Chi Lin*
University of Central Florida, Orlando, Florida, 32816
Daniel E. Schab**
Naval Air Warfare Center Training Systems Division, Orlando, Florida, 32826

Abstract

This paper studies the dead-reckoning equations used in Distributed Interactive Simulation. Dead-reckoning equations of orders zero, one, two, and three are derived. These equations are then used in the dead reckoning of a test flight trajectory. The update rates against various thresholds are used to compare the effectiveness of the equations. The conclusions are: (1) For the dead-reckoning equations of order one and two, the most effective equations are the ones that utilize the latest update information on position, velocity, and acceleration. (2) The update rate reduces as the order of the dead-reckoning equation increases. However, the update rate differences between the second-order and third-order equations for position, and between the first-order and second-order equations for orientation are relatively small. Hence, a combination of second-order equation for position and first-order equation for orientation is a better choice.

1. Introduction

Distributed Interactive Simulation is a synthetic environment within which humans may interact through simulations at multiple sites networked using compliant architecture, modeling, protocols, standards, and databases. DIS is an exercise involving the interconnection of a number of simulators in which the simulated entities are able to interact within a computer generated environment. The simulators may be present in one location or distributed geographically. Communications are conducted over a network.

As a simulator models the behavior of a vehicle in real-time, that vehicle's position/orientation is constantly changing. The simulator modeling the vehicle must inform other simulators of these changes so that all of the simulators participating in the exercise can depict the vehicle correctly at its current location. With a large number of simulators sending their position/orientation information, the network traffic load is tremendous. To reduce the communication traffic in the network, the position/orientation of each entity is not sent through the network every single time a change occurs. Instead, a technique called "dead-reckoning" is used. This term, borrowed from navigation, means establishing the position of a ship by using an earlier known position and then estimating time and motion since that position. Simulators may use dead reckoning to extrapolate the position/orientation of vehicles, thus reducing the frequency of which they would have to obtain the actual information from the network.

2. Dead Reckoning

Dead reckoning is used in the following manner[1]. Each simulator is responsible for maintaining a detailed model of the state of its own vehicle and a precise notion of its own position/orientation over time. In addition, each simulator also maintains a simple dead reckoning model of the state of all other vehicles with which it might possibly interact. The dead-reckoning model is maintained until such time as a new information package, Protocol Data Unit (PDU), is received from the network. According to the proposed DIS standard[2], the Entity State PDU contains, among other information, the entity's position, velocity, acceleration, orientation (Euler angles), and angular velocity. The dead-reckoning model is calculated using this information.

This approach implies that each simulator is also responsible for issuing a new PDU of its own appearance whenever it is necessary. To do this, each simulator must maintain, in addition to its high fidelity model, a dead-reckoning model that corresponds to the model that other simulators are maintaining of its vehicle. After each update of its high fidelity model and its dead-reckoning model, the simulator compares the exact appearance of its vehicle with the extrapolated appearance and issues a new PDU to the network only when the error exceeds a threshold. Figure 1 illustrates this concept.

There are only a few pieces of literature that discuss the dead-reckoning equations. References 3 and

*Assistant Professor of Aerospace Engineering
AIAA member
**Aerospace Engineer, Fighter Attack Branch
AIAA member

4 discussed only the principle of dead reckoning; these discussions did not include detailed analysis or experiment on the dead-reckoning equations. References 5 and 6 used a five-minute segment of an F-16A flight trajectory to experiment with dead-reckoning equations. However, only the equations up to the second order and only one equation for each order were tested. References 7, 8, and 9 studied 15 different dead-reckoning equations. The equations used in these papers came directly from the fixed-step numerical integration formulas. However, since there are two different step times in dead reckoning, as will be discussed in the next section, there is a need for a complete study on the dead-reckoning equations.

In this paper, dead-reckoning equations of orders zero up to three are first derived. These equations are then used in the dead reckoning of a sample flight trajectory. The update rates against various thresholds are used to compared the effectiveness of the equations.

3. Dead-Reckoning Equations

Dead reckoning is a procedure of extrapolation which uses current or previous state variables (position, velocity, and acceleration) to predict the future position. All the extrapolation formulas and integration algorithms are candidates for dead reckoning. However, most of the formulas derived for the fixed step time situation are not suitable for dead reckoning. In the dead-reckoning process, there are two different step times, as shown in Figure 2. In the figure, h is the dead reckoning step time, and Δt's are update step times, where $\Delta t_{-1} = t_0 - t_{-1}$, $\Delta t_{-2} = t_{-1} - t_{-2}$, etc., t_0 is the time of current update, t_{-1} is the time of the last update, and t_{-2} is the time of the update before the last update, etc.. In general, $h \leq \Delta t$'s and $\Delta t_{-1} \neq \Delta t_{-2} \neq \ldots$, etc..

The dead reckoning of the three position variables ($x \ y \ z$), and three Euler angles ($\phi \ \theta \ \psi$) can be executed separately. Any one of these variables exceeding a defined threshold can trigger an update. In the following derivation, variable x is used to represent any one of the six variables.

The dead-reckoning equations can be derived using the Taylor series expansion assuming that $h \ll 1$ and all Δt's $\ll 1$[†]. The order of each dead-reckoning equation is defined as the order of its truncation error minus one. For example, the zero-order equation is given by

$$x_i = x_0 \tag{1}$$

with truncation error

$$v_0 ih$$

where x_i is the dead-reckoning x (position) at time $t_i = t_0 + ih$, $i = 1, 2, \ldots$, which counts the dead-reckoning time steps from the update time, and x_0 and v_0 are the x and $\dot{x}$ (velocity), respectively, at time t_0. The zero-order equation represents the case of no extrapolation. The vehicle remains at its position until receiving the new position update.

The first-order equation is given by

$$x_i = x_0 + v_0 ih \tag{2}$$

with truncation error

$$a_0 (ih)^2/2$$

where a_0 is the $\ddot{x}$ (acceleration) at time t_0. An alternative formula for the first-order equation is given by

$$x_i = x_0 \left[1 + \frac{ih}{\Delta t_{-1}}\right] - x_{-1} \frac{ih}{\Delta t_{-1}} \tag{3}$$

with truncation error

$$a_0 \frac{(ih)^2 + ih\Delta t_{-1}}{2}$$

where x_{-1} is the update position at the last update time t_{-1}. The first-order equation represents the original definition of dead reckoning in navigation.

In DIS, since the Entity State PDU provides the vehicle's acceleration, the second-order equation, given by

$$x_i = x_0 + v_0 ih + a_0 (ih)^2/2 \tag{4}$$

can be used in the dead reckoning of position variables. The second-order equation has more alternatives than the first-order equation, as given by the following equations:

$$x_i = x_0 + v_0 \left[ih + \frac{(ih)^2}{2\Delta t_{-1}}\right] - v_{-1} \frac{(ih)^2}{2\Delta t_{-1}} \tag{5}$$

$$x_i = x_0 \left[1 - \frac{(ih)^2}{\Delta t_{-1}^2}\right] + x_{-1} \frac{(ih)^2}{\Delta t_{-1}^2} + v_0 \left[ih + \frac{(ih)^2}{\Delta t_{-1}}\right] \tag{6}$$

$$\begin{aligned} x_i &= x_0 \left[1 + \frac{ih}{\Delta t_{-1}} + \frac{(ih)^2 + ih\Delta t_{-1}}{\Delta t_{-1}(\Delta t_{-1} + \Delta t_{-2})}\right] \\ &- x_{-1} \frac{(ih)^2 + ih(\Delta t_{-1} + \Delta t_{-2})}{\Delta t_{-1}\Delta t_{-2}} \\ &+ x_{-2} \frac{(ih)^2 + ih\Delta t_{-1}}{\Delta t_{-2}(\Delta t_{-1} + \Delta t_{-2})} \end{aligned} \tag{7}$$

For third-order position dead-reckoning equations, the time derivative of acceleration, $\dot{a}_0$, is not available from the update PDU, therefore extrapolations from the previous updates are required for all cases. Some examples are given by the following equations

$$x_i = x_0 + v_0 ih + a_0 \left[\frac{(ih)^2}{2} + \frac{(ih)^3}{6\Delta t_{-1}}\right] - a_{-1} \frac{(ih)^3}{6\Delta t_{-1}} \tag{8}$$

[†] The dead-reckoning equations will be valid even for large Δt's, since in this situation the dead-reckoning trajectory matches the true trajectory closely.

$$\begin{aligned} x_i &= x_0 + v_0\left[ih - \frac{(ih)^3}{3\Delta t_{-1}^2}\right] + v_{-1}\frac{(ih)^3}{3\Delta t_{-1}^2} \\ &+ a_0\left[\frac{(ih)^2}{2} + \frac{(ih)^3}{3\Delta t_{-1}}\right] \end{aligned} \tag{9}$$

$$\begin{aligned} x_i &= x_0\left[1 - \frac{3(ih)^2}{\Delta t_{-1}^2} - \frac{2(ih)^3}{\Delta t_{-1}^3}\right] \\ &+ x_{-1}\left[\frac{3(ih)^2}{\Delta t_{-1}^2} + \frac{2(ih)^3}{\Delta t_{-1}^3}\right] \\ &+ v_0\left[ih + \frac{2(ih)^2}{\Delta t_{-1}} + \frac{(ih)^3}{\Delta t_{-1}^2}\right] \\ &+ v_{-1}\left[\frac{(ih)^2}{\Delta t_{-1}} + \frac{(ih)^3}{\Delta t_{-1}^2}\right] \end{aligned} \tag{10}$$

$$\begin{aligned} x_i &= x_0 \\ &+ v_0\left[ih + \frac{2(ih)^3 + 3(ih)^2(2\Delta t_{-1} + \Delta t_{-2})}{6\Delta t_{-1}(\Delta t_{-1} + \Delta t_{-2})}\right] \\ &- v_{-1}\frac{2(ih)^3 + 3(ih)^2(\Delta t_{-1} + \Delta t_{-2})}{6\Delta t_{-1}\Delta t_{-2}} \\ &+ v_{-2}\frac{2(ih)^3 + 3(ih)^2\Delta t_{-1}}{6\Delta t_{-2}(\Delta t_{-1} + \Delta t_{-2})} \end{aligned} \tag{11}$$

$$\begin{aligned} x_i &= x_0\left[1 + \frac{(ih)^3}{\Delta t_{-1}^3}\right] - x_{-1}\frac{(ih)^3}{\Delta t_{-1}^3} \\ &+ v_0\left[ih - \frac{(ih)^3}{\Delta t_{-1}^2}\right] \\ &+ a_0\left[\frac{(ih)^2}{2} + \frac{(ih)^3}{2\Delta t_{-1}}\right] \end{aligned} \tag{12}$$

For the orientation dead reckoning, the Entity State PDU provides Euler angles and angular rates. Since the angular acceleration is not available, all dead-reckoning equations that need acceleration information can not be used. The qualified candidates are Eqs. (1) - (3), (5) - (7), and (10) - (11).

The above equations are by no means the complete set of dead-reckoning equation candidates. They are chosen in this study because their truncation errors are smaller than other equations. A more complete list of the dead-reckoning equations can be found in Reference 10.

4. Network Load

The major purpose of using dead reckoning in DIS is to reduce the network traffic. Hence, the first criterion to judge a dead-reckoning equation is how many PDUs it will generate, or, from another angle, how much traffic load it can reduce under certain accuracy requirement.

The flight trajectory chosen to test the network loads of the dead-reckoning equations listed in the previous section is a 35-second flight trajectory with a 40 Hz update rate; this flight trajectory was recorded from a high-fidelity F-16 flight simulator when the pilot was engaged in an Air Combat Maneuvering (ACM) training exercise. Figure 3 shows the three-dimensional plot of this trajectory. Figures 4 to 9 show, respectively, the time histories of the position variables x, y, z and orientation variables (Euler angles) ϕ, θ, ψ.

The flight trajectory has an update rate of 40 Hz. Without dead reckoning, the simulator has to send out an entity state PDU in every 0.025 seconds to provide the updated information on its position and orientation. There are 1400 PDUs in 35 seconds.

The dead-reckoning equations discussed in the previous section are used on the flight trajectory with several different thresholds. The number of PDUs generated in each case is recorded. First, only the dead reckoning of position is considered. In this case, the orientation variables are updated only when the position variables need to be updated. Table 1 shows the number of PDUs generated when the zero, first, second, and third-order dead-reckoning equations, respectively, are used. The numbers in the parentheses following the number of PDUs are the average PDU rates in 35 seconds. From Table 1, it can be seen that the use of dead reckoning reduces the number of PDUs drastically.

The numbers in Table 1 show that (1) The second-order dead-reckoning equations generate significantly less PDUs than the first-order equations. (2) The third-order dead-reckoning equations are comparable with the second-order equations. Among the second-order dead-reckoning equations, Eq. (4) generates the least PDUs.

Now, consider only the dead reckoning of orientation of the aircraft. In this case, the position variables are updated only when the orientation variables need to be updated. Table 2 shows the number of PDUs generated when the zero-, first, second, and third-order dead-reckoning equations, respectively, are used. Similar to the cases in the position dead reckoning, orientation dead reckoning can reduce the number of PDUs. However, since angular acceleration is not available, first-order equations generate about the same level of PDUs as the higher order equations in the orientation dead reckoning. Between the two first-order dead-reckoning equations, Eq. (2) generates the least PDUs.

In practical applications, it is more likely that dead reckoning is applied to both position and orientation variables. Table 3 shows the numbers of PDUs generated by some combinations of dead-reckoning equations. From these numbers, the combination of second-order equation, Eq. (4), for position and first-order equation, Eq. (2), for orientation gives one of the best results. When the orders of the dead-reckoning equations increase, the results are about the same, as shown in the last row of Table 3 (which represents the combination of third-order equation, Eq. (9), for position and second-order equation, Eq. (6), for orientation).

The above results are based on the dead reckoning of one particular test trajectory. Additional ACM maneuvers, where the aircraft is engaged in aggressive changes in orientation such as those associated with a guns tracking solution for fixed wing and rotary wing aircraft, may yield different conclusions. The point is that selection of the proper dead reckoning algorithm will also depend on the role the entity plays in the combat scenario. Further experiments that include a variety of maneuvers will be required.

5. Selection of DR Equations

The following is a numerical example of choosing dead-reckoning equations based on network load requirement.

Example 1 *Assume the standard T1 1.544M bps data line is used in the DIS network. An Entity-State PDU has a minimum size of 1,152 bits.* [2] *Hence in the ideal case, the network can transmit Entity-State PDU at a maximum rate of 1,340 PDU/sec.*

In a 300-entity DIS exercise, each entity, in average, can have a PDU rate up to 4.47 PDU/sec. Assume that the thresholds are set as six feet for position, and two degrees for orientation, and all the entities are maneuvering with a similar complexity as the trajectory shown in Figure 3. From Table 3, there are three cases with their PDU rates under this limit. The case that uses Eq. (2) for both position and orientation gives an average rate of 3.7 PDU/sec which is below the 4.47 PDU/sec limit. However, in practice, this value is too close to the ideal limit, and the network is very likely to saturate in certain occasions. A more conservative choice will be the combination of Eq. (4) for position and Eq. (2) for orientation (2.0 PDU/sec), or other equations of higher order.

In this paper, the authors choose orientation dead reckoning equations from the same set of equations used for position. An alternative approach, proposed by Burchfiel[2,11], is a rotation matrix method based on the concept of Quaternions. This formula assumes the entity moves in a uniform manner (i.e.; constant rate turns, ω = constant), and is given by

$$[\mathrm{DR}] = \mathbf{I}\cos\theta - [\mathbf{a}\times]\sin\theta + \mathbf{a}\mathbf{a}^{\mathrm{T}}(1-\cos\theta) \quad (13)$$

where

[DR]	dead-reckoning rotation matrix
$\mathbf{I}$	unity matrix
$\theta = \mid\omega\mid ih$	angle of rotation in dead reckoning
$\mathbf{a}$	unit vector of rotation axis

$$[\mathbf{a}\times] = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

$$\mathbf{a}\mathbf{a}^{\mathrm{T}} = \begin{bmatrix} a_x a_x & a_x a_y & a_x a_z \\ a_y a_x & a_y a_y & a_y a_z \\ a_z a_x & a_z a_y & a_z a_z \end{bmatrix}$$

Since many aircraft simulations model coordinated turns (i.e., without sideslip), this technique may result in lower PDU rates. However, while this equation may be more accurate, this method in general needs much more computer time due to the additional computational expense of matrix multiplications. Table 4 shows a comparison, where the computation times are scaled values. The rotation matrix method generates less PDUs (21 compared with 25), however, requires about 12 times the computation time.

Assessing computation time is much more complicated than just counting the arithmetic operations in the above dead-reckoning equations. For example, Eq. (2) compared with Eq. (3) generates not only less PDUs but also requires less arithmetic operations to compute. It seems that Eq. (2) is better than Eq. (3). However, there is another factor that may affect the comparison. In the Entity State PDU, there is a field which specifies whether the acceleration and velocity are in the body-axis coordinates or in the world coordinates [2]. In other words, the velocity and/or acceleration received from the PDU could be in either body-axis or world coordinates. If they are body-axis coordinates, a coordinate transformation will be needed to convert them to the world coordinates. For example, when Eq. (2) is used, if the velocity received from the PDU is in the body-axis coordinates, it needs to be converted to the world coordinates. There is no such need, however, if using Eq. (3). Usually, coordinate transformation is much more expensive in computation than the dead-reckoning equation itself. Hence, Eq. (2) needs longer computation time than Eq. (3) in this case. To choose between Eq. (2) and (3), a balance between the network load and computation load depends on the applications.

In Tables 1 - 3, the numbers of PDUs (with the PDU rates in the parenthesis) are listed against the thresholds. As the threshold increases, the number of PDUs generated decreases. From the consideration of network load, a large threshold is preferable. However, a large threshold represents not only large error, but also creates jitters in the dead-reckoning trajectory, which in turn reduces the visual fidelity.

6. Conclusions

From the data presented in the last section, the following observations can be made:

- In general, the higher the order of the equation, the less PDUs it generates. The differences in the number of PDUs are great between equations of order zero, one, and two for the position dead reckoning; the differences are also great between equations of order zero and one for the orientation dead reckoning. Beyond that, the numbers of PDUs generated are about the same.
- The equations which do not need extrapolation from the variables of the previous updates, such as Eqs (2) and (4), generate less PDUs than the equations of the same order that need extrapolation.

If the network load is the only criterion to choose the dead-reckoning equation, any combination of 2nd-order (or higher order) equation for position and first-order (or higher order) equation for orientation is a very good choice. In general, higher order equations need more computation time. If the simulator has a severe limitation in computation power, it has to use a lower order equation, provided that the network traffic load is not a problem. Another factor is that the higher order equations (third-order in position, second-order in orientation) need extrapolation from the variables of the previous updates. This can cause larger errors when there is disturbance in the data. Based on these considerations, the combination of Eq. (4) for position and Eq. (2) for orientation is a better choice.

Acknowledgment

The research reported in this article is sponsored by Defense Modeling and Simulation Office (DMSO) under Department of Defense and administered by the Naval Air Warfare Center Training Systems Division (NAWCTSD) under Contract No. N61339-92-C-0110.

References

[1] Pope, A. R., "The SIMNET Network and Protocols", Report No. 7627, BBN Systems and Technologies, Cambridge, MA, June, 1991.

[2] *Proposed IEEE Standard Draft: Standard for Information Technology - Protocols for Distributed Interactive Simulation Applications, Version 2.0, Third Draft*, Technical Report No. IST-CR-93-15, Institute for Simulation and Training, University of Central Florida, May, 1993.

[3] Saunders, R., "Formal Expression of Dead Reckoning: Mathematical and Representation Recommendation", *Summary Report of the Fifth Workshop on Standard for the Interoperability of Defense Simulations*, Orlando, FL, September, 1991, pp. A-133 A-138.

[4] Lin, K. C., "Dead-Reckoning in Distributed Interactive Simulation," *Proc. of the Third Annual Conference of the Chinese-American Scholars Association of Florida*, Miami, FL, June, 1992, pp. 1 - 5.

[5] Harvey, E. P., Schaffer, R. L., "The Capability of the Distributed Interactive Simulation Networking Standard to Support High Fidelity Aircraft Simulation," *Proc. of the 13th Interservice/Industry Training Systems Conference*, Orlando, FL, November 1991, pp. 127-135.

[6] Schaffer, R., Waters, R., "Dead Reckoning Algorithms and the Simulation of High Performance Aircraft", *Summary Report of the Fourth Workshop on Standard for the Interoperability of Defense Simulations*, Orlando, FL, March, 1991, pp. 233 244.

[7] Morris, K. D., "Long Haul Data Transmission System (LHDTS) Results," Northrop Report No. NOR 91-28, February 1991.

[8] Goel, S. C., Morris, K. D., "Dead Reckoning for Aircraft in Distributed Interactive Simulation," *Proc. of the 1992 AIAA Flight Simulation Technologies Conference*, Hilton Head Island, SC, August, 1992, pp. 277 284.

[9] Goel, S. C., Morris, K. D., "Techniques for Extrapolation, Delay Compensation, and Smoothing with Preliminary Results and an Evaluation Tool," *Summary Report of the Fifth Workshop on Standard for the Interoperability of Defense Simulations*, Orlando, FL, September, 1991, pp. A-47 A-66.

[10] Hoffman, L. L., Lin, K. C., "An Experimental Design for Detecting Optimal Simulated Flight Performance on Distributed Interactive Systems, " Technical Report No. IST-CR-93-16, Institute for Simulation and Training, University of Central Florida, May, 1993.

[11] Burchfiel, J., "The Advantages of Using Quaternions Instead of Euler Angles for Representing Orientation", White Paper ASD 91-001, Third Workshop on Standards for the Interoperability of Defense Simulations, Orlando, FL, August, 1990.

Table 1: No. of PDUs (PDU/sec) vs. threshold, dead reckoning on position only.

Eq.	Order	60’	24’	12’	6’
(1)	0	214 (6.1)	487 (13.9)	837 (23.9)	1315 (37.6)
(2)	1	35 (1.0)	56 (1.6)	79 (2.3)	112 (3.2)
(3)	1	49 (1.4)	77 (2.2)	108 (3.1)	152 (4.3)
(4)	2	16 (.46)	23 (.66)	30 (.86)	39 (1.1)
(5)	2	22 (.63)	29 (.83)	38 (1.1)	49 (1.4)
(6)	2	20 (.57)	28 (.80)	35 (1.0)	49 (1.4)
(7)	2	29 (.83)	39 (1.1)	50 (1.4)	64 (1.8)
(8)	3	15 (.43)	19 (.54)	25 (.71)	31 (.89)
(9)	3	14 (.40)	19 (.54)	25 (.71)	30 (.86)
(10)	3	16 (.46)	20 (.57)	30 (.86)	38 (1.1)
(11)	3	19 (.54)	24 (.69)	31 (.89)	38 (1.1)
(12)	3	15 (.43)	17 (.49)	23 (.66)	31 (.89)

Table 2: No. of PDUs (PDU/sec) vs. threshold, dead reckoning on orientation only.

Eq.	Order	20°	10°	5°	2°
(1)	0	41 (1.2)	77 (2.2)	149 (4.3)	320 (9.1)
(2)	1	17 (.49)	27 (.77)	36 (1.0)	62 (1.8)
(3)	1	27 (.77)	38 (1.1)	51 (1.5)	81 (2.3)
(5)	2	21 (.60)	31 (.89)	44 (1.3)	60 (1.7)
(6)	2	22 (.63)	28 (.80)	39 (1.1)	58 (1.7)
(7)	2	29 (.83)	41 (1.2)	50 (1.4)	79 (2.3)
(10)	3	29 (.83)	35 (1.0)	50 (1.4)	69 (2.0)
(11)	3	29 (.83)	37 (1.1)	50 (1.4)	69 (2.0)

Table 3: No. of PDUs (PDU/sec) vs. threshold, both position and orientation.

Eqs.		Orders		60’&20°	24’&10°	12’&5°	6’&2°
P	O	P	O				
(1)	(1)	0	0	218 (6.2)	489 (14.0)	839 (24.0)	1315 (37.6)
(2)	(1)	1	0	48 (1.4)	85 (2.4)	155 (4.4)	323 (9.2)
(2)	(2)	1	1	41 (1.2)	64 (1.8)	88 (2.5)	129 (3.7)
(4)	(2)	2	1	25 (.71)	34 (1.0)	49 (1.4)	71 (2.0)
(9)	(6)	3	2	26 (.74)	35 (1.0)	46 (1.3)	67 (1.9)

Table 4: PDU rate and computer time of rotation matrix method.

Eqs.		Orders		thresholds = 60’&20°	
P	O	P	O	PDUs (PDU/sec)	computation time
(4)	(13)	2	1	21 (.60)	12
(4)	(2)	2	1	25 (.71)	1

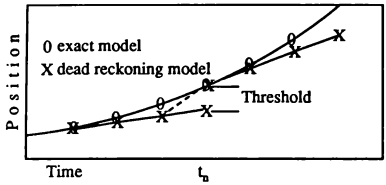


Figure 1: Dead reckoning model.

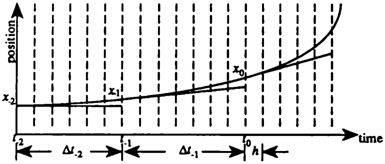


Figure 2: Update time in dead reckoning.

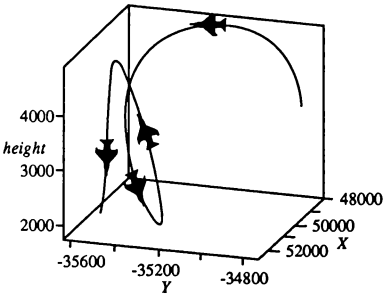


Figure 3: Test flight trajectory.

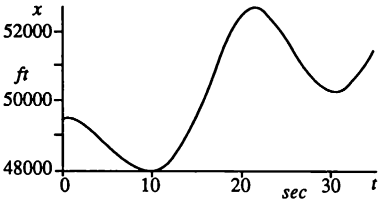


Figure 4: Test flight trajectory, x vs. t.

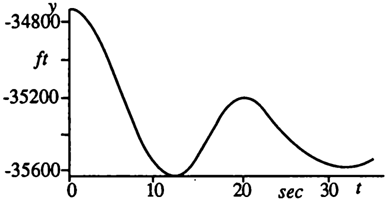


Figure 5: Test flight trajectory, y vs. t.

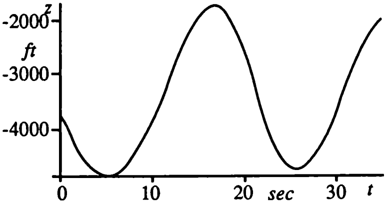


Figure 6: Test flight trajectory, z vs. t. (Note: z is the negative of *height* in Figure 3)

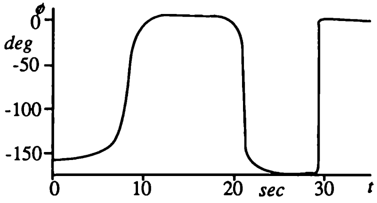


Figure 7: Test flight trajectory, ϕ vs. t.

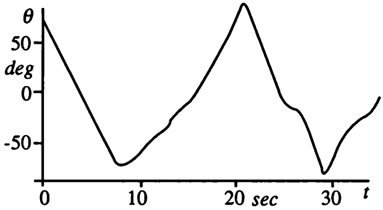


Figure 8: Test flight trajectory, θ vs. t.

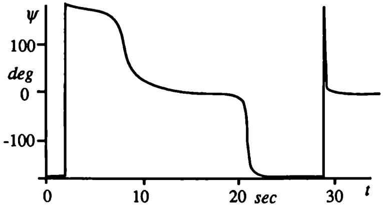


Figure 9: Test flight trajectory, ψ vs. t.

AIAA-94-3426-CP

Somnambulistic Reckoning © for Distributed Interactive Simulations

Don Bernard
Northrop Grumman Corporation
Business and Advanced Systems Development (BASD)
Karl Forsstrom
Northrop Grumman Corporation
B-2 Division

Abstract

Northrop is pursuing a line of research into increasing the effective data bandwidth of real-time, interactive networked simulations by applying unique data filtering techniques over and beyond the traditional Distributed Interactive Simulation (DIS) Dead Reckoning (DR) algorithms. The new filtering technique relies on the concept of transmitting pointers (representing error signals relative to Dead Reckoning deadbands) to statistically generated look-up tables (representing characteristic aircraft behavior). *This work has lead to a method for quadrupling the network bandwidth while reducing average network error and eliminating network overload.*

Background

Northrop is in the process of designing a large local area network for real-time, interactive simulations. We hoped for, and have gotten, a significant improvement in network communication performance through the evolution of new technology. However, this same computer evolution has increased our computation capability and new requirements have rushed in to fill the expanded capacity. Current multi-processor/memory port cluster design has eliminated many of the old communication problems, but until someone overcomes the speed of light limitation, moving an enormous amount of data from one computer cluster to another still poses a problem. The situation is exacerbated by the trend to connect many remotely located laboratories into one large multi-role simulation.

Even with Distributed Interactive Simulation (DIS) Dead Reckoning (DR) and data filter techniques, we may or may not be able to support Northrop's current plans. We are considering a network with a 1 gigabit/sec bandwidth and may have to resort to some network topology that allows for parallel data transfer. A complicated topology scheme brings with it a host of other problems. Limits on budget and computer real estate, synchronization issues and the need to route data between laboratories (with associated latency) come to mind.

While past network designs were adequate at the beginning of a project, later additional requirements eventually burdened the network to the point that it would suffer an occasional overload. This overload condition is usually caused by the simulation reacting to some entity (player) leaving or entering the scenario (missile launch, radar lock, etc.). Although the overload condition only last for a frame or two, it is precisely these moments that interest our analysis people. During an overload condition, access to the network is on a first come first served basis. It is just as likely that some uninteresting or semi-static data (e.g., some avionics mode change) will pre-empt important dynamic data (e.g., missile/target closing velocity). Models that require rate information degrade considerably when any dynamic data is delayed or lost. Can we design a network that would insure high dynamic data transportation?

Because of all these conflicting facts, we started some time ago to take a hard look at our actual simulation data. Specifically, how much of the data was really dynamically changing and how much of the data retained the same old pattern of 1's and 0's?

Nascence of the Concept

Last year Northrop's Systems Simulation Laboratory (SSL) was involved with the US Navy's China Lake test range in an exercise that introduced in-flight aircraft (live assets) into the Northrop Multiple Engagement Simulation (MES) scenario in Hawthorne, CA. Three separate sessions, each one hour long, successfully demonstrated the feasibility of using live assets along with simulated participants in an air combat force-on-force simulation environment.

A by-product of this exercise was about six hours of F-18 and A7 aircraft and simulation data from three different sources: 1) telemetry (down-link) data from the live F-18's and A-7's at China Lake, 2) raw network modem data (China Lake to Hawthorne) and 3) simulation data in Hawthorne. All data packets were time tagged by Global Positioning System (GPS) receivers on the live aircraft and in the Hawthorne Laboratory. With this unique set of data available, we decided to investigate the statistical characteristics of the network data in order to gain understanding of distributed simulation data transmission problems and to improve network capability.

We limited our data set to six aircraft state variables (x, y, z, φ, θ, ψ). The other data were not of interest for our feasibility study or were considered too noisy for analysis.

To simulate the China Lake-to-Hawthorne network we used two side-by-side Silicon Graphics, Inc., (SGI) 4D/80GT workstations (WS) with a VME-to-VME connection and an Ethernet link. Workstation 1 (WS1) provided either China Lake F-18 raw data or simulated F-18 data from our own pilots and engineers in Hawthorne to Workstation 2 (WS2) through a simulated network (Ethernet). The network data transmission involved floating point to integer conversion on WS1 prior to getting on the network and an integer to floating point conversion at the receiving WS2. The basic DIS Dead Reckoning algorithms were used with error tolerances set for each aircraft state variable. These algorithms determine if a state variable is transmitted on the network based on a minimum deviation from the previous position (state).

Each workstation displayed simple 3-D stick airplane models to present an overview and comparison of the "raw data" aircraft dynamics to the "network degraded" dynamics. In addition, the workstations displayed real-time computed statistical analysis including state variable error ("raw data" minus "network degraded data"), histograms of frequency of error to magnitude of error, Root Mean Square (RMS) of errors for each state variable, etc. See Figures 1 and 2 for a time domain and frequency domain representation.

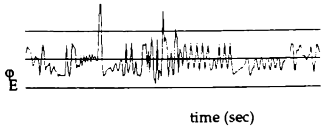


Figure 1. RMS Bank Angle Error

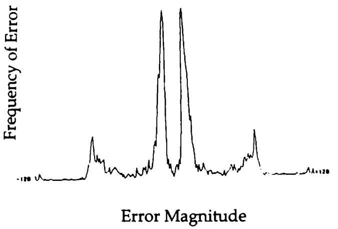


Figure 2. Frequency/Magnitude Histogram

The "Aha Reflex"

We were in the process of quantifying factors that influence simulation fidelity, e.g., RMS error versus "human guts feel," when we noticed that one of our statistical measurements, the frequency of error to magnitude of error histogram, appeared to show unique distribution characteristics for each of the six state variables. We suspect this distribution is made up of pilot corrections required for flight course maintenance (inner pair of peaks) and course maneuver (the outer two peaks). The implication was that some network bits were working harder than others!!

This realization triggered a search for a method to improve the efficiency of the floating point to integer (and back to floating point) conversion process. Why waste all those useless bits on data that had inherently very small errors?

Method

Because the technique relies on a combination of 1) historical time-varying simulation data exhibiting unique frequency distribution characteristics and 2) current time-varying simulation data, the method has been compared to that of a sleep-walker navigating around his house in the dark; i.e., the somnambulist's (sleep-walker's) basic knowledge of location of doors, halls and rooms is modified and corrected occasionally after bumping into forgotten or unexpected obstacles. Hence the term Somnambulistic Reckoning (SR) has been suggested to differentiate this method from Dead Reckoning (DR) where future position is simply extrapolated from known previous positions. The SR method requires thorough statistical (frequency) analysis of previously gathered simulation data in addition to the current time-varying data.

The SR method is similar in concept to the Japanese Morse code implementation: an index (number) is transmitted in place of the origional word or phrase; this number is used to look up entries that are numerically indexed in a standard dictionary.

The SR method devised concentrates precision in the range where conversion is done most often. A non-linear (modified exponential: $Ke^{-|x|}$) array of 256 floating point words was generated with spacing (precision) approximating the distribution of our error histograms and a gap around zero error for a lower limit threshold (the conventional DR deadband). This array was used in a table lookup scheme to produce an index (1 byte = 2^8 = 256) which is sent over the network instead of the usual number. Note that a one byte index of a simulation parameter, e.g., bank angle error relative to DR deadband, was transmitted rather than a floating point number (or integerized version of a floating point number). The index is saved for use in the next conversion.

The next conversion is computed from the table lookup by taking the difference between a new floating point number and the floating point number from the array pointed to by the previously saved index.

This successive approximation feedback process is necessary to eliminate accumulation errors. We will later show that this feedback process is also useful when data need not be transmitted.

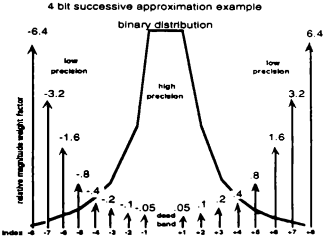
Our future Northrop application will amplify on the examples used in this paper. We anticipate having higher rates in our future simulations. The China Lake simulation didn't use missiles and the data was noise filtered on the aircraft. We will probably have to use a 12 bit word on selected data to accommodate these higher rates. This added complexity will require us to set up classes of data objects such as missile class (high precision) and ground target class (low precision) much like the DIS Protocol Data Units (PDU's).

To further reduce the burden on the user, systems calls to read and write a data block will be developed. The idea behind all of this is to make the transfer of data totally transparent to the user while improving performance.

Simple Example

The following simple four bit successive approximation analog to digital converter algorithm illustrates this process. The converter works by sampling an analog input signal and storing a digital representation of that signal as a data word of a specified number of bits. Four bits implies $2^4 = 16$ possible bins to store data. Assume that these 16 bins are given index numbers from -8 for a negative units to +8 for a positive units and that the bins have been "loaded" with a predetermined set of floating point unit values.

As an example, an instantaneous input signal of 5.333 units falls between index(+7) = 3.2 and index(+8) = 6.4 and therefore the index value 7 is saved. Next the input value and the stored floating point value for index(+7) is used to compute an error: 5.333 - 3.2 = 2.133. This error falls between index(+6) and index(+7) and becomes a new input to compute a new error: 2.133 - 1.6 = 0.533. The process continues and produces a series of index values (7,6,4,2) and a residue (dead band error) of 0.033. In summary, an arbitrary input value can be characterized or described by a series of index numbers and knowledge of the distribution (statistics) that created the values referenced by the index numbers. As in standard DIS Dead Reckoning, if the instantaneous input signal is within the deadband (±0.05), no data is transmitted.



step	error	index	value
1	5.333	7	3.2
2	2.133	6	1.6
3	0.533	4	0.4
4	0.133	2	0.1
	0.033	dead band	

Figure 3. Example of Successive Approximation

The above example had to iterate four times to reach the dead-band solution area. For our network data transmission application, we need to increase the precision of the process and determine a distribution curve that will converge to the dead band area on the first approximation for a large majority of the conversions. Our search for this tailored distribution curve (Figure 4) was done by analyzing our live asset simulation data. We made the necessary changes in the analysis program and then ran all of our flight data through the new method to produce new statistical data (RMS error was used as the metric).

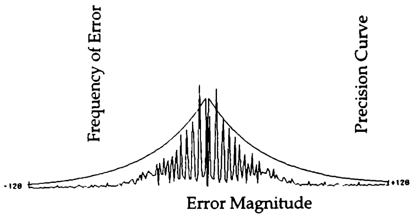


Figure 4. Tailored Distribution Curve

Distribution Curves

Prior to testing, our flight data base was cleansed of obvious down-link noise errors then searched to find periods of maximum dynamic data. Culling this data identified two other sets of characteristic distributions: landing/takeoff and navigation. These were not investigated because of their relatively low dynamics. The F-18 and A-7 data were similar showing attitude aircraft roll (φ) to be the predominant dynamic factor. Hence, roll was used to establish a maximum attitude data. A second distribution was used for position data.

A decaying exponential distribution gave the best results. Additional tests were conducted to find a dead band that gave the best trade-off between RMS error and number of transmissions.

Figure 4 (same input data as Fig. 2) shows how the non-linear precision function (smooth line) affects the error (jagged line) distribution in the conversion process. The successive approximation function and a second order extrapolation scheme [1]tend to corral the majority of conversions into the high precision area.

Retesting with New Distributions

Final testing with the two workstations produced expected results. Distribution histograms became flatter, instantaneous errors stayed within tolerances, RMS errors decreased some, but the *network data rates went down by a factor of four!*

New statistical analysis indicated a great deal of redundancy in data transmission. In dynamic combat maneuvers, index values for all six state variables do not change all at the same time and in normal flight, relatively few updates are necessary.

Prioritization and Network Budget

With this information, a second data compression method was applied [2]. Relying on the feedback process to accommodate accumulation errors, we eliminated superfluous data transmissions by generating a flag word where each bit in that word indicated when transmission was necessary because of changed data. The data following the flag word is uniquely identified by the bit map in the flag word as seen in Figure 5. Applying this compression method to our raw modem data indicated that this method alone could have saved 45% of network bandwidth on our original China Lake tests.

The methods described above allows us to give priority to data with the most error and *eliminate the overload situation on the important data!!* A "not to exceed bandwidth budget limit" is established for each computer cluster on the simulation net. The budget is based on an algorithm that takes into account total network bandwidth, confidence level of accuracy, knowledge of vehicle dynamics (tank versus fighter) and the history of a local and global fidelity monitor, i.e., RMS error.

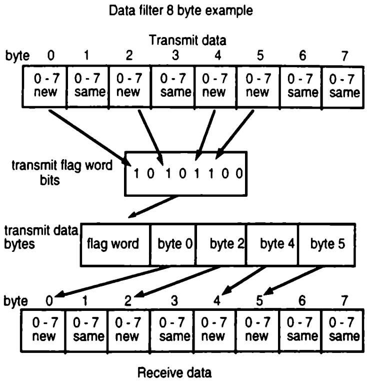


Figure 5. Data Filter to transmit only new data

Currently this budget parameter is transmitted to all network participants during simulation initialization. However, we are considering a dynamic budget allocation scheme which would alter the budget throughout the simulation exercise. The total networked simulation system is, in some sense, self regulating by limiting transmissions to the highest total error (instantaneous + accumulated). The system will sacrifice short term error for precious network throughput much like an auto pilot reacting to large dynamic inputs, tolerating instantaneous errors and then eliminating error in later transmissions.

"Poor Man's Kalman Filter"

A serendipitous side effect was also discovered. Our network communication with China Lake ground station ran flawlessly, but occasional noise bursts on the radio down link caused some problems. Our old dead reckoning software would accept nonsense data as truth and a noise burst would try to send the vehicle hundreds of miles out of the state. Our new design has a built in sanity check. When given absurd updates, our system will transmit the largest update it knows to be reasonable making it behave like a poor man's Kalman filter.

Data Integrity

A second less obvious spin off from our investigations comes from the statistical software package that we have developed and the lessons we have learned from this experience. Past simulation data analysis strategy has been focused on customer requirements. An unhappy simulation customer/analysts could argue that a perceived simulation anomaly was due to the simulation process not their model (and they might be right!). We have had independent checks on the overall and relative timing, but until now we had no handle on data integrity. Expanding our analysis tool set and the feedback process of learning from using that set should help in understanding our complex, parallel, multi-cluster, multi-laboratory communication environment.

Caveats

Designs developed in a perfect environment must also work in the real, error prone world. This technique is not fault tolerant. Therefore, methods such as those described above should not be considered in a system where noise is a predominant factor. However, in our tests the hardware worked well, i.e., no parity errors or checksum errors were detected. This is not to suggest a system should run "open loop." If data is transmitted both ways in a network, error detected at the receiving station (running parity and checksum) would signal activation of a repair algorithm.

Conclusion

By the time this paper is published, Northrop expects to have four cluster (each containing four 100+ MHz processors) on a fiber optic network of some tens of meters. Later in the year, the network will grow to sixteen clusters and an assortment of standalone processors separated by even greater distances. In short we will be faced with the formidable task of maintaining and synchronizing an enormous amount of data.

The simple two node tests using our China Lake simulation data has provided us with several useful techniques and tools.

Specifically we have learned to transform a vehicle floating point state vector (6 x 32 = 192 bits) to a byte oriented vector (6 x 8 = 48 bits) and to filter this vector and all other data. *The resulting effective increase in network bandwidth can exceed 400% without any change of hardware.* By additionally prioritizing this filtered data, we can insure the timely transmission of important data and reduce or eliminate overload of dynamic network data.

A side benefit is a collection of tools and techniques for monitoring network performance. We should be able to extrapolate our findings and get a much better estimate our new network's capability and integrity.

References

1. S. Goel and K. Morris; "Dead Reckoning for Aircraft in Distributed interactive Simulation"; AIAA Flight Simulation Technologies Conference, Hilton Head, August 1992, p277.

2. D. Cohen; "The Next Generation of DIS PDU"; White paper presented to DIS; Orlando Fla.
March 1994.

AIAA-94-3427-CP

SHARED-MEMORY NETWORKS (SMNs): A KEY ENABLING TECHNOLOGY FOR DISTRIBUTED REAL-TIME SYSTEMS

George J. Valentino
Senior Technical Manager

SYSTRAN Corporation
4126 Linden Avenue
Dayton, Ohio 45432-3068
USA

AIAA Flight Simulation Technologies Conference
Scottsdale, AZ
1 - 3 August 1994

ABSTRACT

Real-time systems, such as test instrumentation, hardware and person in-the-loop simulators, virtual reality environments, aircraft avionics suites, and process control systems have evolved significantly during the last few years, keeping pace with new technologies and attempting to provide the required speed and bandwidth interconnectivity among the various sensors, processors, and other nodes which comprise these systems. To support such applications, Shared-Memory Network (SMN) technology provides an attractive alternative to the more conventional approach which utilizes single or multiple CPU's contained within the same chassis. This presentation will describe the history, technology, current trends, and selected applications of shared-memory networks in real-time simulation, testing, and computational applications. More recent applications of this technology includes the use of SMNs as the networking basis for Virtual Reality (VR) systems, and as the real-time network of choice for local, real-time 'player' networks connected to the geographically distributed Distributed Interactive Simulation (DIS) network.

SYSTEM ARCHITECTURES

In order to place SMN technology in perspective, first consider several other processor and memory system architectures: distributed memory, shared- memory, and distributed shared-memory.

Distributed Memory.

A distributed shared-memory architecture is illustrated in Figure 1.

In this architecture each processor node has access to a local private memory. Communications occurs between processor nodes. The contents of each local memory is usually unique.

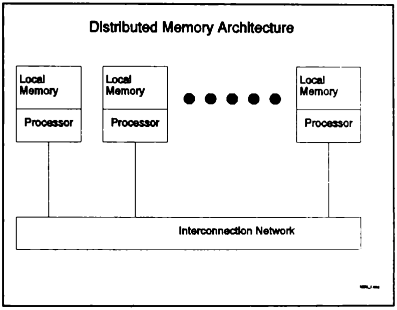


Figure 1 Architecture 1: Distributed Memory

Shared-Memory.

A shared-memory architecture is illustrated in Figure 2. In this architecture, each processor and each memory module is individually connected to the communications network. Any processor can access any memory module, depending upon the needs of the individual application being processed. Again, the contents of each memory module is usually unique.

Distributed Shared-Memory.

A distributed shared-memory architecture is illustrated in Figure 3. In this architecture, processor and memory modules are each individually connected to the communications network and, local communi-cations between a processor node and a memory module is also established. Again, the contents of each memory module are usually unique.

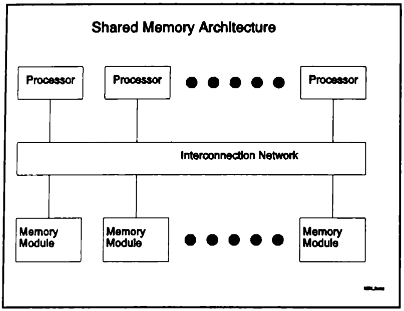


Figure 2 Architecture 2: Shared Memory

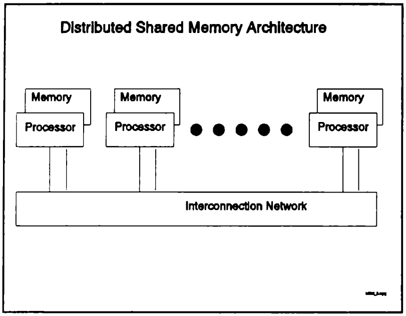


Figure 3 Architecture 3: Distributed Shared Memory Architecture

Replicated Shared-Memory.

A replicated shared-memory network (SMN) architecture is illustrated in Figure 4. In this architecture, the memory modules (not the processor nodes) are the system elements connected to the communications network. And, unlike the three previous architectures, the contents of each memory module are the same (i.e., replicated). The benefits of this architecture to real-time systems is the basis of this paper.

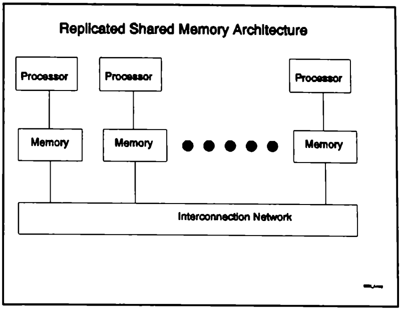


Figure 4 Architecture 4: Replicated Shared-Memory Network (SMN)

SMNs: SOLUTIONS TO MANY NEEDS

In the mid-1980's, in an attempt to provide a means with which to simulate, monitor, and test evolving avionics architectures and specific aircraft "suites", the concept of replicated shared-memory networks (SMNs) was developed [1,2,3]. Conceptually, a replicated shared-memory network is rather straightforward -- each node on the network utilizes a local, transparent copy of network memory that is continuously refreshed with new data values, whether computed at the local node, or from any other node on the network. The resultant logical model of the network includes all nodes accessing a single, global, multi-port memory. As a consequence of this logical model, several distributed processing nodes can perform real-time computations with computational power equivalent to, or in excess of, that available from a single, massive, high-performance computer.

Message-passing networks, such as Ethernet or FDDI (fiber distributed data interface), are best suited for data processing applications, which are more concerned about I/O and bandwidth than low data latency. Based on the OSI/ISO seven-layer architecture, they require too much system overhead and network software to meet the speed requirements of real-time networks.

Replicated shared-memory networking offers an elegant solution to the real-time, computer-to-computer communications problem required in real-time simulation, monitoring, and testing systems. A SMN reduces multiple-computer systems to one virtual computer, diminishing system development and maintenance costs. It features both system-level architectural simplicity and ultra-fast (low latency) data communications. Suited to problems requiring more than one computer, it offers data transfer at the speed of light.

The real-time speeds, logical global memory, ability to connect dissimilar processors, and high-speed processor fiber-optic connectivity of SMNs are the very features which are specified when SMN technology is included in real-time systems. SMN technology provides a simple, yet elegant, way to fulfill several computing needs. Figure 5 illustrates some of these needs: to amass computing power, to connect specialized computers, or to connect remote facilities.

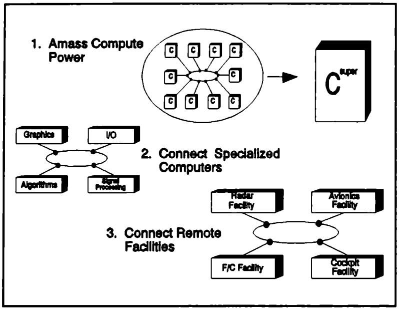


Figure 5 Shared-Memory Network Technology Can Support A Variety of Requirements

The traditional method used to amass computing power would be to acquire a bigger computer. Micro- and mini-computers are usually replaced with super-mini and super-computers. SMN technology permits supercomputer performance through the utilization of a network of relatively simple individual processors. A ring of multiple processors (Figure 5, Item 1) can provide a substantial amount of computing; and, this computing power can be easily adjusted in increments much finer, and more economical, than integer numbers of supercomputers.

Another way that SMN technology excels is the ability to interface and connect several specialized computers, as illustrated in Figure 5, Item 2. Since SMN technology provides a transparent hardware interface to each individual computer, via additional replicated local memory, various types of SIMD, MIMD, and other individual computer nodes can be easily interfaced. The system designer must insure that unique parameter names and addresses are maintained throughout the overall system.

The ability to connect remote facilities is another capability of SMN technology (Figure 5, Item 3). In many Government and Commercial campus settings, specialized analysis, processing, simulation and/or testing facilities are located in different buildings, sometimes separated by hundreds to many thousands of meters. SMN technology is being used to connect these facilities in order to conduct real-time operations over local and global distances.

SHARED-MEMORY NETWORKS: DEFINITION

A replicated SMN is a remarkably simple concept to understand. All network processing nodes are equipped with identical network cards that contain additional computer-addressable memory. This memory is configured into the overall address space available to all user programs. The additional memory at each node is known as "replicated shared-memory" because it contains replicas of all data to be shared among system computers.

Relative offset addresses to all replicated shared-memory data are the same at all network nodes, so all shared-memories in the computer system literally "replicate," or are identical to, each other. If a computer word in one network node changes, the word and its address are automatically passed on to the same address in all other replicated shared-memories in the system. Since no software is needed to assemble, pass, accept, and decompose the message as it travels from computer to computer, the complete updating of all replicated shared-memories in the system occurs in microseconds.

While each computer physically has its own memory, the results are essentially the same as if all computers were attached to a common memory. The distributed computer system appears at all times as one large "virtual" multi-processor to the real-time application, because the interprocessor communications occur at memory speeds.

Figure 5 illustrates the major concepts behind SMN technology. Physical and logical diagrams are illustrated in Figures 6 and 7, respectively.

Figures 8 and 9 provide an additional example of how SMNs are used. In Figure 8, the overall memory map for the example system is portrayed. This memory has been partitioned into six regions, each to hold a specific class of system parameters. These parameters are associated by being of a particular class (i.e., Class A, Class B,...,Class F). It is important to note that each class of parameters is computed and written to the memory by a single procesing node within the system. These nodes are denoted Node 1 through Node 6. While only individual nodes are responsible for computing and writting the results of unique applications to these regions, the data that resides in any particular region is being used by multiple nodes within the system. For example, the Class A data, which

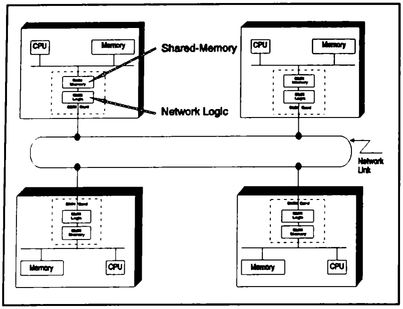


Figure 6 The Physical Implementation of a SMN

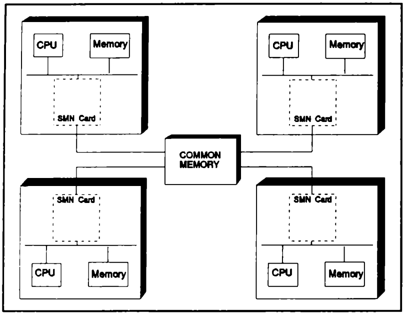


Figure 7 Logically, a SMN Shares a Single Common Memory

are computed by Node 1, are themselves used (in this example) by Nodes 3, 5, and 6. Class B data are used by Nodes 1 and 4. The other examples are as listed in the figure.

Another way of portraying these example relationships is illustrated in Figure 9. Here the logical flow of information is conveyed from each source node (NO arrowhead) to each sink node (with arrowhead). Please note that Figure 9 does NOT represent the wiring diagram for this example system, but only the logical flow of the information. When implemented with a SMN, the six nodes in Figure 9 would be linked via a ring network, with each node containing a physical copy (i.e., a replication) of the entire shared-memory map.

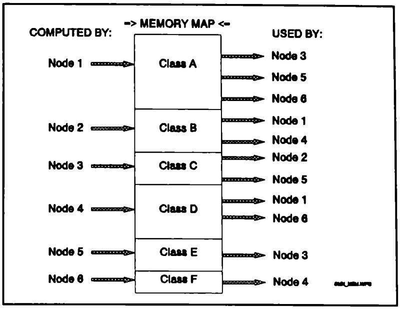


Figure 8 Memory Map for an Example SMN

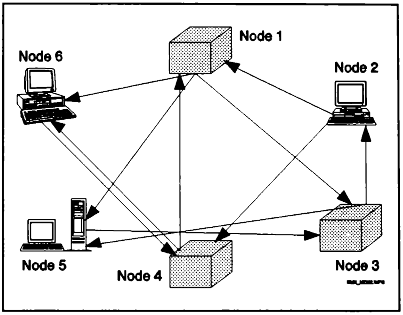


Figure 9 Logical Flow of Information

SHARED-MEMORY NETWORKS: EXAMPLES

SMN technology has been used in a number of real-time and distributed computing applications in facilities across the U.S., in Europe, and in Japan. A limited sample of these applications are briefly described below.

MISILAB.

SMN technology was used as a cornerstone of the Eglin AFB, FL MISILAB (Missile Simulation Laboratory) to support a hardware-in-the-loop simulation with a frame rate of 1000 Hz [11]. In this installation, an Encore 32/8780 functioned as the primary simulation executive control system, processing flight dynamics algorithms, performing environmental modeling, and computing synthetic-line-of-sight (SLOS) and flight table computations. A SUN 4/330 workstation, the RF Control System, hosted multiple i860 processors operating as dedicated high-speed target and

ECM (Electronic CounterMeasures) model compute engine. A Silicon Graphics model 4D/310 workstation, operating in read-only mode, monitored SCRAMNet^{TM} (SYSTRAN's Shared Common RAM Network) data traffic during hardware-in-the-loop simulations to yield high-resolution realistic displays of the air-to-air engagement scenario.

The EISE Facility.

SMN technology was also utilized within the Sacramento Air Logistic Center's EISE (Extendable Integrated Support Environment) as the A-10 ISF (Integrated Support Facility) [9]. The EISE completely simulated the aircraft's actual flight environment, its aerodynamics, and its on-board sensor complement. For other aircraft subsystems, the EISE utilized actual flight equipment included in an avionics hot-bench. SCRAMNet was selected for use within EISE because of its ability to provide ultra-high speed and low data latency. In addition, the replicated shared-memory network was flexible and able to interconnect multiple vendor's computers. The configuration of the EISE facility is illustrated in Figure 10.

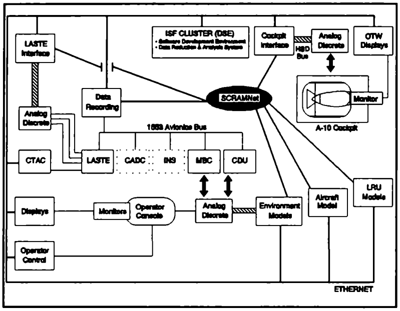


Figure 10 SMN EISE Configuration

The CAVE.

The CAVE (CAVE Audio Visual Experience Automatic Virtual Environment) was a Virtual Reality (VR) system conceived and demonstrated by the Electronic Visualization Laboratory (EVL) at the University of Illinois at Chicago [4]. The objective of the CAVE designers was to develop a VR system in which the human participant, or participants, were surrounded by the virtual image space. The CAVE is a 5-sided room with each side envisioned as a rear-projection screen for a Silicon Graphics Crimson VGXT workstation-driven Electrohome ECP 4100 projectors. The current CAVE configuration uses only four projection surfaces, since a material strong enough for use as the floor is still being sought. The EVL utilized SCRAMNet SMN technology as the network backbone for the CAVE. See Figure 11 for an artists conception of how the 5-sided CAVE would be configured.

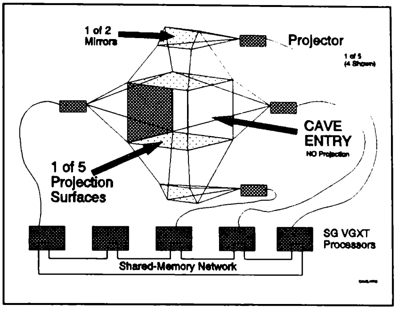


Figure 11 Major Components of the CAVE

VHDL Simulator.

The current method for designing VHSIC (Very High Speed Integrated Circuits) ASICs (Application Specific Integrated Circuits), FPGAs (Field Programmable Gate Arrays) and WSI (Wafer Scale Integration) is through the use of VHDL (VHSIC Hardware Design Language), a hardware specification language which permits simulations of the complex electronics prior to the fabrication of the silicon hardware chips. Because some of the individual chips have upwards of 250,000gates, simulation times can be excessive.

An effort is underway [5] in which SMN technology is being used to connect several Silicon-Graphics and SUN workstations into a high speed, computational load-balancing processor array.

DIS CAU.

US DOD simulators are now being designed to interface to one another via DIS (Distributed Interactive Simulation) protocols. A major training simulator corporation is using SCRAMNet SMN products to interconnect 'local' simulators to one another (within the same 'cell') and to a CAU (Cell Adapter Unit) which acts as a gateway to other DIS-compliant gateways and simulation cells. The CAU processor retrieves data from its local copy of shared-memory and converts these local data into formats that conform with DIS PDU (Protocol Data Unit) standards. Figure 12 illustrates the concept behind using SMN technology in local, real-time DIS 'player' networks connected to DIS exercises.

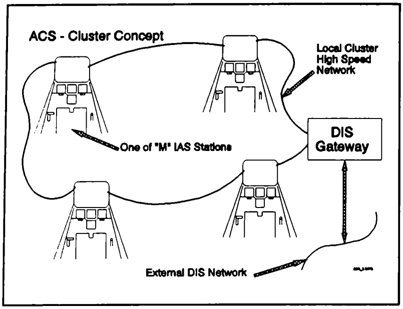


Figure 12 Local Real-Time HITL and/or PITL 'Players' Within a DIS Exercise

The Wright-Laboratory Real-Time Network

A vision is being implemented, within the Wright Laboratory, at Wright-Patterson AFB, OH, which is the seamless integration of major real-time simulation facilities into a single, composite R&D simulation, test, and experimentation facility. This integration of facilities is progressing in an incremental fashion, with the connection of some Flight Dynamics Directorate and Avionics Directorate simulation facilities now underway. Later additions to this simulation network will include the remaining real-time facilities in those Directorates, with the addition of complementary facilities in other Wright-Patterson AFB organizations. A notional view of how SMN technology is supporting the Wright Laboratory real-time network is illustrated in Figure 13.

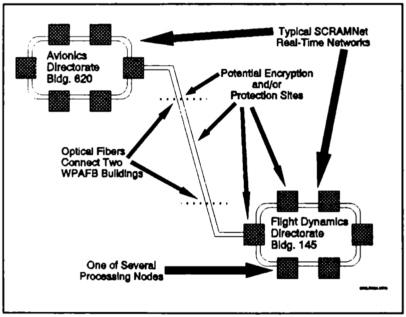

Typical SCRAMNet Real-Time Networks
Avionics Directorate Bldg. 620
Potential Encryption and/or Protection Sites
Optical Fibers Connect Two WPAFB Buildings
Flight Dynamics Directorate Bldg. 145
One of Several Processing Nodes

Figure 13 Classified Data Must Be Protected or Encrypted to Support Wright Laboratory Distributed Simulations

AVIONICS AND SMNs

The on-board avionics networks of both military and commercial aircraft represent one of the most complex real-time networking environments that any new networking and computational paradigm will be subjected to. This environment has been selected as the basis in this paper for illustrating how the attributes of SMN technology can be applied to this most challenging application area.

Many existing (e.g., F-15, F-16, B-1, B-2, C-5, C-17) and evolving (e.g., F-22 and RAH-66) aircraft utilize avionics with embedded computer communication protocols that impose significant CPU overhead to initiate and receive network or bus traffic. In addition, the software required for task/data synchronization and for constructing these complex "message passing" protocols is very complicated, thus increasing software development and maintenance costs. Computers, when connected into networks to perform real-time functions, typically need application-to-application transmission latencies of no more than 50 to 500 microseconds. Standard avionic bus protocols are marginally successful in meeting data transmission requirements.

SMN technology provides alternatives to standard avionics architectures, for use in retrofitting existing aircraft during out-year maintenance, or for use in new platforms (either airborne or spaceborne). SMNs use physical copies of the system memory at each computational node to produce a system with a single, logical global memory. This logical model permits each computation node, with no software overhead, to write data into or read data from memory, in real-time. Data transmission speed is performed by hardware with no software "overhead" and is incredibly fast, on the order of 250 to 800 nanoseconds (0.25-0.60 microseconds) per node. Standard avionic bus protocols are 10 to 100 times slower. SMN technology can provide for on-board, real-time, avionics communication between current and emerging avionics processor designs.

Avionics represents the largest single cost of today's aerospace systems. The retrofitting of existing systems or the development of new systems will also require a similar proportion of overall funding. The use of SMNs for avionics provides an architecture unconstrained by processor type, processor speed, and backplane configuration. Additional benefits include improved LRU communication via improved network access using a hardware implemented network protocol (i.e., with no software overhead); "simple" software/hardware coupling; less expensive and more easily maintainable software due to processor-transparent inter-processor communications.

Embedded computer communication protocols currently in use for avionics impose significant CPU overhead to initiate and receive network traffic. Also, the software required for task/data synchronization and for constructing bus messages is very complicated, thus increasing software development and maintenance cost. Also, if the sophisticated data synchronization software of these current-day systems does not attempt to evenly distribute the bus transmissions over the frame computational cycle, then frame overruns will likely occur. One reason for this is due to the tendency of programmers to perform data inputs at the beginning of a computational frame and data outputs at the end. Such practices result in zero network bandwidth utilization until the end of the frame. Then suddenly the network is saturated, and data traffic can be skewed across the next frame-start time.

Application-to-application transmission times for commonly available bus and LAN network systems (e.g., MIL-STD-1553 and HSDB) can range from 100 microseconds to many milliseconds. When real-time application programs are executing at frame rates from 10 to 100 cycles per second (typical in avionics) such latency is a severe design problem - if not intolerable.

As a result of the complex communication protocols now used in avionics, task reconfiguration in current avionic hardware and software architectures is very cumbersome. In these systems specific CPUs must be designated as "back-up" processors for specific tasks. All data associated with a critical ("backed-up") task must be transmitted to the back-up CPU, possibly doubling the network loading. Also, all the CPUs involved must switch master bus lists to the alternate configuration and the entire system resynchronized. This traditional reconfiguration process is difficult and may actually require many seconds to effect. In a real-time combat situation, seconds may not be available.

All the problems outlined above can be largely obviated with the use of replicated shared-memory network technology. Replicated shared-memory is a special networked memory communication card installed into the backplane of the host processor. Host CPU read and write cycles are monitored by the shared-memory card. When the physical shared-memory is written into by the host CPU, the networked shared-memory communications card automatically transmits the changed data to share or "replicate" the changes on all other host CPUs in the network. Therefore, every host CPU on the network has a shared-memory "reflection" of the same memory.

Replicated shared-memory does not necessarily dictate other network features. For example, the physical communication link could be a ring, bus, or star. Furthermore, the network may utilize any of the popular network protocols—including CSMA/CD, token, and polling. What we emphasize is that "inter-processor" communication can be accomplished as simply and efficiently as "intra-processor" communication, i.e., with the speed and efficiency of a memory access. This is the advantage of a shared-memory networking approach.

Several significant benefits can be derived from shared-memory as a communications technology. One benefit is that the host-node CPU overhead for interprocessor communications is reduced to zero. This is because all network communication functions—i.e., encode, decode, send and receive—are accomplished in network hardware and independent of host CPU intervention. In avionic systems, where every millisecond of CPU time if very significant. This one benefit may save 10-30% of the CPU and memory capacity of each avionics computer.

Another advantage to shared-memory networking is the increased effective utilization of the network bandwidth. This increased efficiency is due to two basic concepts which are natural to replicated shared-memory systems. The first concept is the data filtering feature that true replicated shared-memory systems may implement. It has been demonstrated that avionic and flight control systems only change a maximum of approximately 25% of the data they transmit in any given minor-frame time. To capitalize on this fact, it is natural for a replicated shared-memory network to transmit one-data-word messages, given the discrete, random access nature of memory storage and retrieval. By designing the replicated shared-memory hardware to transmit only "changed" data, a replicated shared-memory network needs to only transmit one-fourth (25%) of the data that other networks have to transmit.

A second concept giving replicated shared-memory systems a natural advantage has to do with leveraging the utilization of the network bandwidth across the full frame time of the avionic compute cycle. A network which automatically transmits each data item as it is updated by the host CPU will tend to distribute the transmission of data across the entire frame time. This is in contrast to current protocols which "bunch" the data into packets at the beginning and end of frame times and thus waste much of the link bandwidth.

Many special avionics applications, such as aircraft test instrumentation, special sensor suites, and on-board simulation and exploitation devices, can easily benefit by utilizing SMN technology. The use of an application-to-application gateway to interface a SMN special application network to an existing 1553 avionics multiplex bus is one way that this technology can enhance avionics processing payoff. This concept is illustrated in Figure 14.

VIRTUAL REALITY IS REAL-TIME

During the last several years, the public has

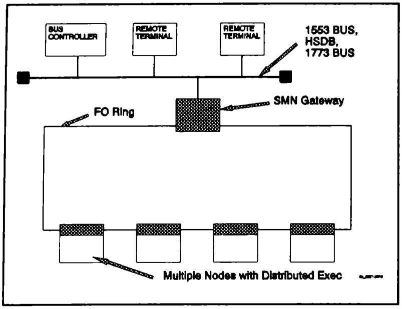


Figure 14 Using Special Gateways, SMNs May Be Interfaced to Conventional Avionics Buses

witnessed an explosive growth in the marketing, reporting, and application of VR. Several dozen vendors now offer a wide array of VR appliances and components, such as body suits, data gloves, articulated and head-mounted display systems, and other input and output devices. These components are complemented with an equally large array of software environment applications that support the fabrication of VR domains limited only by the imagination of the developer. These domains may be as common as landscapes, building facades, building interiors, and other architectural items. More unique domains may range from the interior of matter to the reaches of the universe. Some domains may exist using constructs not possible by existing physical laws.

It is my opinion, however, that independent of the number and sophistication of the input/output devices, and independent of the domain of application, VR can be defined rather simply:

"Virtual Reality is Real-Time simulation."

If this definition troubles some readers, perhaps the "simulation" aspect of the definition should be removed, so that the definition becomes:

"Virtual Reality is Real-Time."

If this definition, or at least hypothesis, of VR is accepted, then it would seem that some of the technologies used to support other real-time systems may be applicable to the computational needs of VR systems. As SMN technology has become a cornerstone of many real-time system applications, it is my belief that SMN technology is directly applicable to "large" VR systems.

There are several techniques commonly employed in order to achieve the response times demanded by real-time Hardware-in-the-Loop (HITL) and/or Person-in-the-Loop (PITL) simulations. (HITL and/or PITL systems are always real-time systems or simulations. In fact, either HITL and/or PITL systems/simulations may require several different real-time response loops, such as a 1000 Hz loop for a missile guidance unit, a 100 Hz loop for an aerodynamic model, and a 60 Hz loop for a visual display system.) These techniques include:

(1) Use the computer with the most MIPS (or Dhrystones, or other metric)

(2) Use a computer with multiple internal processors and multiport memory

(3) Use multiple processors and interconnect them with a message-passing local area network

(4) Use replicated Shared-Memory Network technology

Option (1) will eventually fail because at some point even the most powerful individual computer will reach its processing limit. Usually these are expensive individual machines, so the cost to add the next increment of processing capability will be prohibitive (relative to the specific application).

Option (2) is a variant of Option (1), as most of the world's most powerful individual computers utilize multiple processors within their "container" all accessing a multiport memory.

Option (3) is a typical solution that will work in an office environment but will falter or fail in a real-time environment.

Option (4) is the real-time option - it provides a mechanism for processor communications without the software overhead associated with conventional LANs.

If the hypothesis, "Virtual Reality is real-time", is accepted as true, then SMN technology - a real-time technology - is directly applicable to many VR applications. These applications can include various mixtures of people-, hardware-, and algorithms-in-the-loop including all varieties of input and output devices.

A general SMN technology architecture that can be used for a VR system is illustrated in Figure 15. This architecture permits substantial flexibility for growth in the overall system. Initially only two nodes are needed to complete the ring network backbone of the SMN. Once this backbone is established, growth can be achieved in several ways, such as:

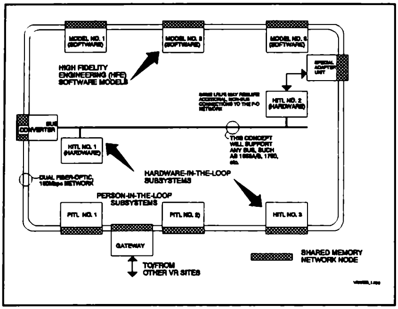

MODEL NO. 1 (SOFTWARE)
MODEL NO. 2 (SOFTWARE)
MODEL NO. 3 (SOFTWARE)
HIGH FIDELITY ENGINEERING (HFE) SOFTWARE MODELS
SPECIAL ADAPTER UNIT
HITL NO. 2 (HARDWARE)
THESE I/Fs MAY REQUIRE ADDITIONAL NON-BUS CONNECTIONS TO THE F-O NETWORK
BUS CONVERTER
HITL NO. 1 (HARDWARE)
THIS CONCEPT WILL SUPPORT ANY BUS, SUCH AS 1553A/B, 1760, etc.
DUAL FIBER-OPTIC, 150Mbps NETWORK
HARDWARE-IN-THE-LOOP SUBSYSTEMS
PERSON-IN-THE-LOOP SUBSYSTEMS
PITL NO. 1
PITL NO. 2
HITL NO. 3
GATEWAY
TO/FROM OTHER VR SITES
SHARED MEMORY NETWORK NODE

Figure 15 General Architecture for a SMN Technology VR System

- Add new functionality to each node, until the capabilities of the node are reached,

- Add additional software (non-input/output) nodes,

- Add additional input/output nodes.

There are various ways to "parallel process" the algorithms needed to implement large and complex VR systems. Many traditional multi-processor and/or LAN approaches to parallel processing force the VR designer to accept the overhead and latency penalties of software protocols. SMN technology represents a method to meet the demanding real-time requirements of VR systems without this overhead or latency while permitting flexibility in the choice and mixture of processors best suited for particular VR algorithms.

It is hoped that these ideas will stimulate the VR community into considering SMN technology for the most intensive VR applications.

RECENT DEVELOPMENTS

SCRAMNET-LX™

In the mid-1980's, SYSTRAN developed SCRAMNet®, the well-known Shared Common Random Access Memory Network that has enjoyed success throughout Government and Industry. SCRAMNet® incorporated several features that allows it to excel in real-time, distributed network implementations. These features are summarized in Figure 16.

■ Shared-Memory Communications

■ 250 nSec/node Transport Delay

■ 6.5 MByte/Sec Data Throughput

■ Fiber Optic Media

■ Up to 2 MByte Memory Size Options

■ Many Bus Options

Figure 16 SCRAMNet® Features

To stay at the forefront of SMN technology, SYSTRAN has continually listened to our customers and incorporated the features they desired into our products. About two-years ago our engineers embarked upon a total redesign of SCRAMNet® to accommodate these features, and to reach our other goals of reduced parts count, smaller product footprint, and overall enhanced performance. The result is SCRAMNet-LX™, which was recently introduced on 25 April 1994. SCRAMNet-LX™ features are summarized in Figure 17. SCRAMNet-LX™ is fully backwards compatible with existing SCRAMNet® installations.

■ Shared-Memory Communications

■ 250 nSec/node Transport Delay

■ 6.5 to 16.7 MByte/Sec Data Throughput

■ Coax or Fiber Optic Media Options

■ Up to 8 MByte Memory Size Options

■ Single-Slot Solution

■ Many Bus Options

■ Custom ASIC Solution

Figure 17 SCRAMNet-LX™ Features

SCRAMNet-LX™ has been designed for a variety of computer buses, including VME (for 3, 6, and 9U chassis), EISA and ISA (both for PCs), GIO (for SGI machines), Sbus (for SUN machines), MicroChannel® (for IBM machines), and IndustryPack®. This array of bus solutions will continue to permit heterogenous processors to be networked together to meet diverse requirements throughout Government and Industry.

The CAMILLE Network^{TM}

SYSTRAN continues to explore new networking solutions. An activity currently underway at SYSTRAN is the development of The CAMILLE Network^{TM}. CAMILLE signifies Computer Architectures using MERLIN Interactive, Low-Latency Events. MERLIN refers to the MEmory Routed, Logical Interconnection Network originally conceived at Sandia National Laboratory [10]. The MERLIN programming model is illustrated in Figure 18.

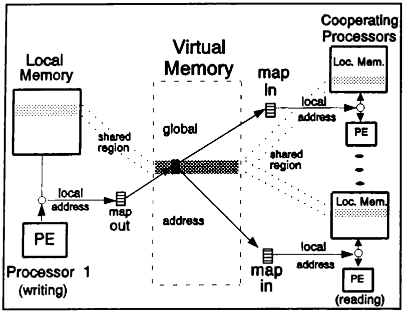


Figure 18 MERLIN Programming Model

The CAMILLE Network^{TM} utilizes the MERLIN paradigm to achieve interactive (i.e., real-time), low-latency (i.e., 250 to 800 nanoseconds), events (i.e., communications) for a variety of computer architectures (i.e., point-to-point or ring, within a rack or across a campus). Design goals for The CAMILLE Network^{TM} are delineated in Figure 19.

- Implement MERLIN Paradigm
- Incorporate COTS Components
- Incorporate Emerging Standards
- Minimize latency
- Incorporate fault-tolerance

Figure 19 The CAMILLE Network^{TM} Design Goals

One of the major applications we see for The CAMILLE Network^{TM} is the interconnection of large numbers of workstations into workstation arrays that together perform as a single machine. This capability is intended to meet the HPCC (High Performance Computing and Communication) applications requirements of researchers in many widlely diverse fields, including fluid dynamics, astronomy, materials processing, and similar computational intense applications. Figure 20 illustrates this clustering concept.

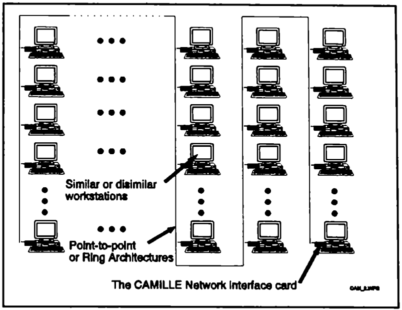

Similar or dissimilar workstations
Point-to-point or Ring Architectures
The CAMILLE Network Interface card

Figure 20 The CAMILLE Network^{TM} Will Interconnect Large Arrays of Workstations Into Powerful Computing Clusters

IndustryPack^{TM} Data I/O Modules

Most real-time systems that SCRAMNet® or The CAMILLE Network^{TM} are targeted for require the acquisition of external system signals and/or the generation of signals to control external systems. SYSTRAN has recently developed a family of data I/O modules based on the IndustryPack^{TM} form factor. These data I/O modules are easily added to any of several carrier cards, such as the Motorola MV-162. A typical application is illustrated in Figure 21.

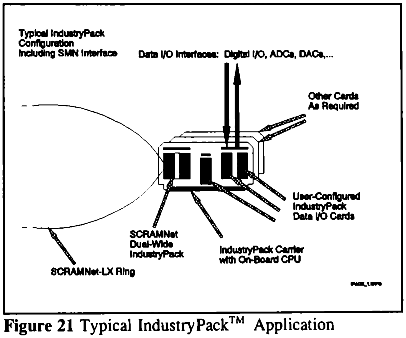

Typical IndustryPack Configuration Including SMN Interface
Data I/O Interfaces: Digital I/O, ADCs, DACs,...
Other Cards As Required
User-Configured IndustryPack Data I/O Cards
SCRAMNet Dual-Wide IndustryPack
IndustryPack Carrier with On-Board CPU
SCRAMNet-LX Ring

Figure 21 Typical IndustryPack^{TM} Application

SUMMARY

Replicated shared-memory networks offer several unique advantages for a broad variety of real-time applications. They are being used throughout government and industry, within the U.S., in Canada, Europe, and Japan. These networks fulfill the specialized requirements of the real-time applications domain.

BIBLIOGRAPHY

[1] McDonald, J.E., "An Architecture for Event-Driven Real-Time Distributed Computer Systems," Proc. of NAECON '83, May 1983.

[2] Hague, D., et.al., "A Data-Driven Operating System for Data-Driven Architecture of Real-Time Systems," WL/AAAF Technical Paper, September 1984.

[3] Warden, G.G. and Hodge, S., "Local Area Networks for Real-Time Simulators and Integrated Support Facility,"SYSTRAN Technical Paper, May 1986.

[4] Cruz-Neira, C., et.al., "The CAVE Audio Visual Experience Automatic Virtual Environment," Communications of the ACM, June 1992/Vol.35,No.6.

[5] Charley, D., et.al., "High Speed Communication for Simulation of Large VHDL Models," Dept. of Electrical Engineering, University of Cincinnati, 21 October 1992.

[6] Valentino, G.J., "Shared-Memory Networks: Description, History, and Candidate as a Future Avionics Architecture," Proc. of NAECON '93, May 1993.

[7] Bohman, T., "Shared-Memory Computing Architectures for Real-Time Simulation - Simplicity and Elegance," presented at Sim-Tec, November 1992.

[8] Stephenson, M.M. and Warden, G.G.,"An Advanced Multi-Purpose Support Environment (AMPSE)," Proc. of NAECON '87, May 1987.

[9] Bollinger, K.W., "The Extendable Integration Support Environment (EISE)," EISE Program Office, SM-ALC/TIEFA, McClellan AFB, CA, 1991.

[10] Maples, C. Sandia National Laboratories, and Whitte, L., SUNY, "MERLIN: A Superglue for Multicomputer Systems," Proc. of COMPCOM '90, February 1990.

[11] Application Note #107, "SCRAMNet Network Enables MISILAB Test Facility To FulFill Its Charge," 1992.

ACKNOWLEDGEMENTS

The author thanks all personnel at SYSTRAN for the information and insights they have provided regarding replicated shared-memory networks. Special thanks to Lynn Trainor, Gary Warden, and Martin Davis for information used in this paper.

INFORMATION

Readers are invited to contact the author for additional SMN technology technical and applications information. He can be reached at the address listed on the title page of this paper. Phone numbers are: Voice (513) 252-5601,or Fax (513) 258-2729. Internet: GVALENTINO@SYSTRAN.COM

EXTRAPOLATION OF AIRPLANE STATES

Amnon Katz* and Kenneth Graham†
Department of Aerospace Engineering
The University of Alabama
P. O. Box 870280, Tuscaloosa, AL 35487-0280
akatz@ua1vm.ua.edu

ABSTRACT

Methods for predicting six degree of freedom airplane states that exploit coordinated flight and other airplane specific assumptions are derived and applied. They include: (1) Extraction of orientation from trajectory with angle of attack correction. (2) Closed form trajectory extrapolation based on constant longitudinal and transverse acceleration. (3) "Phugoid scheme" for six degrees of freedom extrapolation at constant angle of attack in coordinated flight. The new methods prove significantly more accurate than generic second order position, first order orientation methods.

I INTRODUCTION

The state of a rigid body in 3-space involves six degrees of freedom. The motion history of such body therefore involves six arbitrary functions of time. The extrapolation schemes in current use are based mainly on the assumption that these six functions are smooth. Several variants of orientation prediction have been used. Second order extrapolation of position proves better than linear prediction[1]. The schemes apply to a general entity.

In this paper we address the prediction of airplane flight and exploit the knowledge that the entity extrapolated is an airplane. The pilot of an airplane manipulates only four major controls. In coordinated flight he‡ determines only three arbitrary functions. We exploit the decreased freedom for more accurate prediction.

The DIS standard[2] calls for extrapolation of the motion of simulated entities in order to minimize data transmissions over the network. Updates are transmitted only when the prediction method fails to maintain a predetermined error threshold. In this way, a natural metric arises: the frequency of required updates becomes the measure of extrapolation fidelity. The better the prediction of future motion, the less frequent the updates. Our methods provide an improvement by a factor of 2 to 3 over the generic comparison method.

II OVERVIEW OF AIRPLANE SPECIFIC METHODS

Extrapolation in time is prediction. Naturally, it is impossible to predict what a pilot will do. But a pilot does not control six arbitrary functions, he controls only four in the form of his inputs to the throttle, elevator, ailerons, and rudder. If the added assumption is made that coordinated flight be maintained, then only three degrees of freedom remain. Coordinated flight will be assumed from here on.

Three arbitrary functions are exactly what is required to define a space-time trajectory. (A space-time trajectory is the combination of a curve in space and a schedule of the times at which the airplane is at any given point.) An airplane can (subject to performance limitations) fly an arbitrarily specified smooth space-time trajectory; but its orientation while doing so is determined.

To visualize this, imagine the space trajectory being a circle in the horizontal plane and the time schedule specifying that the circle be followed at a given constant rate of speed. The pilot can accomplish this. The load factor and bank are determined by a universal relationship to rate and radius of turn. The pitch attitude is slaved to the angle of attack that produces the requisite load at the given speed. The yaw input is, of course, slaved to the requirement of coordinated flight.

It follows that orientation is defined by the space time trajectory. The determination of body pitch requires detailed knowledge of geometric and aerodynamic data of the particular airplane. On the other hand, the orientation of a *velocity system* aligned with the relative wind ("velocity orientation") for the case of coordinated flight can be *extracted* from the trajectory by a universal algorithm that requires no data specific to the airplane type. This algorithm was first given by one

[*] Professor.

[†] Graduate student.

[‡] The word "he" in all its forms is used here generically to mean "he", "she", or "it".

of us (AK) in a position paper to the sixth DIS workshop[4] †.

In many flight regimes the angle of attack is small, and pitch changes related to angle of attack are difficult to perceive visually. If the system of velocity coordinates is substituted for the body system for the purpose of visual display, the resulting visual error will usually be insignificant.

The converse is not true. Body orientation may not be substituted for velocity orientation for the purpose of extrapolating the motion. An attempt to do so results in significant error. Fortunately, in the DIS context, orientation information is available, an adjustment from the velocity orientation to the body orientation can be made. The difference between the two, for coordinated flight, is the angle of attack. The angle of attack can be determined from the initial data and, if it is assumed constant, it can subsequently be used to "correct" velocity orientation to body orientation. This is "extraction with correction" or EC, which was proposed for the first time in reference 3. We found the angle of attack correction crucial for accurate extrapolation of the data we analyzed.

Extraction of the orientation can be applied together with any method for predicting the trajectory. We try it, for reference, in the context of the assumption of constant earth acceleration, which is the simplest and most commonly used method.

It has long been felt that constant body acceleration is a better assumption than constant earth acceleration where airplanes are concerned. In reference 3 we formulated a variant of this concept in which the components of acceleration are constant in a velocity frame – a frame aligned with the velocity and the transverse acceleration. The result is plane motion that admits closed form expressions[3]. We refer to it as "constant longitudinal and transverse acceleration", or, in brief, CLTA. Extraction of orientation and CLTA are two independent algorithms. The former can be used with any method of trajectory extrapolation. The latter can be used with any method of orientation extrapolation. Put together as CLTA+EC, they make a complete extrapolation algorithm for airplanes.

CLTA may describe airplane flight better than constant earth acceleration. This scheme will provide correct long range predictions in a level turn. However, any pull-up or pushover would be continued into an inside or an outside loop. This is avoided by our third algorithm - phugoid extrapolation[3]. The phugoid scheme offers a complete six degree of freedom extrapolation, with airplane dynamics and coordinated flight built in. Left to its own devices, phugoid extrapolation will stabilize any mild transient condition into a stable level, climbing, or descending turn, or into a straight climb or descent, if initially unbanked.

The phugoid scheme assumes constant thrust, constant angle of attack, and coordinated flight. There are two modes of lateral control: (a) constant bank, and (b) zero rate of roll. The first is appropriate for normal attitudes and maneuvers, the second to aerobatic flight. One mode or the other is selected based on attitude. The phugoid scheme, too, is based on the velocity system, which can be propagated using universal relationships and requires a minimum of vehicle specific data. An angle of attack correction as described above is built in. The premise of constant angle of attack justifies using the AOA determined initially throughout the phugoid prediction.

The phugoid scheme is unable to predict what a pilot will do. It does predict what the airplane will do without conscious input from the pilot. The assumptions of constant angle of attack and constant thrust correspond to no motion on the longitudinal stick or the throttle. Constant bank and zero roll do require control inputs; however, these inputs are "psychomotor" as far as the pilot is concerned. Phugoid extrapolation requires an estimate of the aerodynamic efficiency (L/D ratio). It is not, however, overly sensitive to the estimate. Recall that L/D affects only the damping of phugoid oscillations and not their frequency.

Section III addresses trajectory geometry and kinematics. The scheme for extracting orientation from a given space time trajectory is given in Section IV. Section V presents the closed form expressions for trajectories maintaining constant longitudinal and transverse acceleration (CLTA). The phugoid scheme is defined in Section VI. Section VII describes exceptions to the various algorithms, which arise in special cases, and the work-arounds that we adopted. Section VIII presents the results of applying the airplane specific methods to available data.

† The suggestion that some aircraft characteristics might be exploited in the extrapolation scheme has been voiced occasionally. The use of coordinated flight was mentioned at the 4th DIS workshop, but the algorithm for doing this was not given until the 6th workshop.

III TRAJECTORY GEOMETRY AND KINEMATICS

A space time trajectory is a sequence of positions in 3-space parametrized by time. Geometrically, the space trajectory consists of the same sequence of positions, however parametrized. The natural geometric parameter is the arc length. We denote derivatives with respect to s by a prime and derivatives with respect to t by a dot.

The basic geometric relationships are

$$\vec{r}' = \vec{e}_t, \tag{3.1}$$

where $\vec{e}_t$ is a unit vector tangent to the trajectory.

$$\vec{r}'' = \vec{e}_t' = \kappa \vec{e}_n. \tag{3.2}$$

Here $\vec{e}_n$ is the unit vector along the first normal to the curve, and κ is the curvature, which is also the inverse of the momentary radius of the trajectory. A small segment of the trajectory is approximated by a circular arc of radius $1/\kappa$ lying in the plane defined by $\vec{e}_t$ and $\vec{e}_n$. The second normal to the trajectory, which completes $\vec{e}_t$ and $\vec{e}_n$ to a right handed triad is

$$\vec{e}_2 \equiv \vec{e}_t \times \vec{e}_n. \tag{3.3}$$

Moving from geometry to kinematics, we now turn to the velocity $\vec{v}$, which is the time derivative of $\vec{r}$, and the acceleration $\vec{a}$, which is the second time derivative. In evaluating these we use equations (3.1), (3.2), (3.3) above and the chain rule of derivatives with

$$\dot{s} = v. \tag{3.4}$$

We find

$$\vec{v} \equiv \dot{\vec{r}} = v\vec{e}_t, \tag{3.5}$$

$$\vec{a} \equiv \dot{\vec{v}} = \dot{v}\vec{e}_t + \kappa v^2 \vec{e}_n. \tag{3.6}$$

Thus the velocity has the magnitude $\dot{s}$ and the direction of $\vec{e}_t$; The acceleration consists of a longitudinal component of magnitude $\dot{v}$ in the direction of $\vec{e}_t$, and a normal component being the centripetal acceleration in the direction of the first normal $\vec{e}_n$.

Equations (3.5), (3.6) may serve to define the longitudinal and and normal acceleration, given the trajectory. In the alternative, when velocity and acceleration are given, as in DIS, at the timestamp of a received packet, the tangential direction is given by

$$\vec{e}_t = \frac{\vec{v}}{v}. \tag{3.7}$$

The acceleration may be decomposed into a tangential component

$$\vec{a}_t = (\vec{a} \cdot \vec{e}_t)\vec{e}_t, \tag{3.8}$$

and a normal component

$$\vec{a}_n = \vec{a} - \vec{a}_t, \tag{3.9}$$

which is in the direction of the first normal.

IV EXTRACTION OF ORIENTATION FROM THE TRAJECTORY

We now get to the algorithm for extracting the airplane orientation from the trajectory. The orientation amounts to a definition of the body fixed coordinate system in terms of an earth fixed system. To be definite, we adopt a body system with the x_b axis pointing forward, towards the nose, the y_b axis pointing to the pilot's right towards the right wingtip, and the z_b axis directed down through the airplane's floor.

The extraction procedure actually addresses a different system of axes – the velocity system of axes (also known as "wind axis system"). In this system, the x_v axis is aligned with the airplane's velocity vector, the y_v axis points to the right and is in the plane defined by the x_v axis and the wing span. The z_v axis completes the right hand triad. It roughly points in the direction of the floor of the airplane. We refer to the orientation of the velocity system of axes as "velocity orientation".

The body system is related to the velocity system by a rotation in pitch equal to the airplane's angle of attack and a rotation in yaw equal to the sideslip angle. This last angle vanishes in coordinated flight. In this case, the angle of attack can be determined from the data in the DIS appearance PDU as the angle between the velocity and the x_b axis. A correction by the angle of attack may be applied to the extracted velocity orientation to estimate the true body orientation. We incorporate the correction directly into our phugoid scheme (Section V) and make this correction an option that can be used with the other trajectory extrapolation methods in conjunction with the orientation extraction method which is about to be described. The alternate approach is to neglect the difference between the velocity orientation and the body orientation and assume that, under ordinary conditions of flight, the difference is too subtle to be noticed visually. In this case, one can make do with the velocity orientation and use it in place of body orientation to display remote players.

Denote the unit vectors in the x_v, y_v, and z_v directions respectively by $\vec{i}, \vec{j}$, and $\vec{k}$. We start by aligning the x_v axis with the trajectory

$$\vec{i} = \vec{e}_t. \tag{4.1}$$

This neglects the pitch connected with the current angle of attack. As already discussed in the introduction, the error is usually not noticeable visually. The advantage of this approximation is that it requires no information specific to the particular airplane.

With $\vec{i}$ determined by equation (4.1) there still remains one degree of orientation freedom, namely the rotation of the body system around $\vec{e}_t$. This is done in the transverse plane, that is the plane orthogonal to $\vec{e}_t$. We therefore focus our attention in this plane. The acceleration $\vec{a}$ has already been decomposed into tangential and normal components in (3.8) and (3.9). We similarly decompose the acceleration of gravity $\vec{g}$ into a longitudinal component

$$\vec{g}_t = (\vec{g} \cdot \vec{e}_t)\vec{e}_t, \tag{4.2}$$

and a normal component

$$\vec{g}_n = \vec{g} - \vec{g}_t, \tag{4.3}$$

The balance of normal specific forces is expressed by

$$\vec{l} = \vec{a}_n - \vec{g}_n, \tag{4.4}$$

where $\vec{l}$ is the active normal specific force – in aircraft practice this is the lift divided by the mass. The vectors on the right hand side of (4.4) are defined in equations (3.9) and (4.3). Equation (4.4) in turn defines $\vec{l}$.

The condition of coordinated flight is that $\vec{l}$ be in the direction of the negative z body axis:

$$\vec{k} = -\frac{\vec{l}}{l}. \tag{4.5}$$

The body triad is completed by

$$\vec{j} = \vec{k} \times \vec{i}. \tag{4.6}$$

When the vectors $\vec{i}, \vec{j}, \vec{k}$ are expressed in an earth coordinate system X, Y, Z, the matrix of components

$$M_v = \begin{pmatrix} i_X & j_X & k_X \\ i_Y & j_Y & k_Y \\ i_Z & j_Z & k_Z \end{pmatrix} \tag{4.7}$$

describes the active rotation from earth to velocity coordinates.

If the angle of attack correction is to be employed, the correction may be computed once from the initial data and applied every time the orientation is used for visual presentation. For illustration, assume that orientation is expressed as a rotation matrix M (the active rotation from earth coordinates). Once a body orientation, M_b, and a velocity orientation, M_v, are both available, compute the correction matrix

$$M_{cor} = M_v^T M_b. \tag{4.8}$$

Then, every extrapolation step, after M_v is obtained, M_b may be produced as

$$M_b = M_v M_{cor}. \tag{4.9}$$

An analogous procedure applies to quaternions. Euler angles must be converted to one or the other.

V TRAJECTORY EXTRAPOLATION

There still remains the question of extrapolating the trajectory. In this section this is addressed based on the assumption of components of acceleration being constant either in earth coordinates or in velocity coordinates. Closed form results are presented for both cases.

When earth acceleration is constant,

$$\vec{a} = \vec{A}, \tag{5.1}$$

the resulting trajectory is

$$\vec{r} = \vec{r}_0 + \vec{v}_0 t + \tfrac{1}{2}\vec{A}t^2. \tag{5.2}$$

This is an elementary result that is in any high school physics text. We include it here for completeness.

There aren't many flight maneuvers that maintain constant (non zero) earth acceleration. Keeping the body components of acceleration constant may have a slightly wider applicability. In this case we assume that the tangential and normal components of acceleration in equation (3.6) are both constants. Denoting them by A and B respectively we find

$$\dot{v} = A, \tag{5.3}$$

$$\kappa v^2 = B. \tag{5.4}$$

Equation (5.3) integrates into

$$v = v_0 + At. \tag{5.5}$$

We further suppose that the motion stays in the plane defined by the initial $\vec{e}_t$ and $\vec{e}_n$. Multiply equation (3.2) by $\dot{s} = v$ to find

$$\dot{\vec{e}}_t = v\kappa\vec{e}_n. \qquad (5.6)$$

This identifies the angular velocity of $\vec{e}_t and\ \vec{e}_n$ in their plane as

$$\dot{\theta} = v\kappa, \qquad (5.7)$$

where θ is the angle of $\vec{e}_t$ with an earth fixed x_p axis in the plane of the motion. Now use (5.4) and (5.5) to recast (5.7) as

$$\dot{\theta} = \frac{B}{v} = \frac{B}{v_0 + At}. \qquad (5.8)$$

The last equation integrates into

$$\theta = \frac{B}{A} ln\left|1 + \frac{At}{v_0}\right| = \frac{B}{A} ln\left|\frac{v}{v_0}\right| \qquad (5.9)$$

To complete the integration of the trajectory we must determine the position of the airplane. We do this first in a system of coordinates x_p, y_p in the plane of the motion with the x_p axis along the initial $\vec{e}_t$ and the y_p axis along the initial $\vec{e}_n$. The governing differential equations are

$$\dot{x}_p = v\, cos\theta, \qquad (5.10)$$

$$\dot{y}_p = v\, sin\theta. \qquad (5.11)$$

To proceed from here, divide the last two equations by (5.8) to obtain

$$\frac{dx_p}{d\theta} = \frac{v^2}{B} cos\theta, \qquad (5.12)$$

$$\frac{dy_p}{d\theta} = \frac{v^2}{B} sin\theta. \qquad (5.13)$$

Now rearrange (5.9) as

$$v = \pm v_0 e^{\frac{A}{B}\theta}, \qquad (5.14)$$

and substitute into (5.12), (5.13). The result is

$$\frac{dx_p}{d\theta} = \frac{v_0^2}{B} e^{2\frac{A}{B}\theta} cos\theta, \qquad (5.15)$$

$$\frac{dy_p}{d\theta} = \frac{v_0^2}{B} e^{2\frac{A}{B}\theta} sin\theta, \qquad (5.16)$$

The last two equations are reduced by quadrature to

$$x_p = \frac{v_0^2}{4A^2 + B^2} e^{2\frac{A}{B}\theta}(2Acos\theta + Bsin\theta), \qquad (5.17)$$

$$y_p = \frac{v_0^2}{4A^2 + B^2} e^{2\frac{A}{B}\theta}(2Asin\theta - Bcos\theta), \qquad (5.18)$$

Equations (5.17), (5.18) describe a family of curves that depend on only two parameters, which may be selected as:

$$\alpha \equiv \frac{A}{B}, \qquad (5.19)$$

$$c \equiv \frac{B\, v_0^2}{4A^2 + B^2}. \qquad (5.20)$$

In terms of these, equations (5.17) and (5.18) can be rewritten as

$$x_p = c\, e^{2\alpha\theta}(2\alpha\, cos\theta + sin\theta), \qquad (5.21)$$

$$y_p = c\, e^{2\alpha\theta}(2\alpha\, sin\theta - cos\theta), \qquad (5.22)$$

where c has a dimension of length and serves as a scale factor for the trajectory. α is dimensionless and determines its shape. Each trajectory is a spiral that converges to the origin on one side and spreads out to infinity on the other. Figure 1, a plot of y_p/c against x_p/c, is a sample curve for $\alpha = 0.125$.

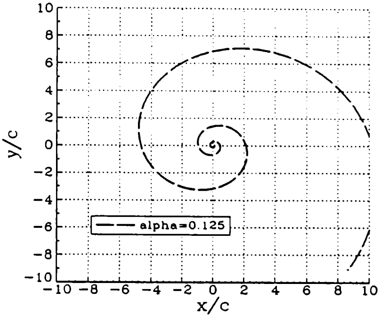


Figure 1: CLTA trajectory ($\alpha = 0.125$).

Each spiral is self similar, and any two points on the spiral are equivalent in the following sense: Any selected point on the spiral may be moved to any other selected point by a similarity transformation, consisting of rotation and dilation, that maps the spiral onto itself.

The velocity vector, $\vec{v}$, and the position vector, $\vec{r}$, relative to the earth system origin are given by

$$\vec{v} = v(cos\theta\vec{e}_{t0} + sin\theta\vec{e}_{n0}), \tag{5.23}$$

$$\vec{r} = \vec{r}_0 + (x_p - 2\alpha c)\vec{e}_{t0} + (y_p - c)\vec{e}_{n0}. \tag{5.24}$$

where $\vec{e}_{t0}$ and $\vec{e}_{n0}$ are the unit vectors in the initial tangential and normal directions, respectively. The full closed form procedure is:

1. Determine the initial parameters: $\vec{e}_{t0}$, $\vec{e}_{n0}$, v_0, θ_0, A, and B from the initial data.
2. Compute c and α (equations (5.19), (5.20)).
3. Compute v in terms of t (equation (5.5)).
4. Calculate θ in terms of v (equation (5.9)).
5. Find x_p, y_p in terms of θ (equations (5.21), (5.22)).
6. Obtain $\vec{v}$ and $\vec{r}$ (equations (5.23), (5.24).

The derivations above assume that $A \neq 0$ and $B \neq 0$. Should B vanish, then, from (5.8), θ is a constant, the velocity frame does not rotate, and its motion is rectilinear. The equations for constant acceleration in earth coordinates apply. (The case that both A and B vanish is included.) If $A = 0$, but $B \neq 0$, the trajectory is a circle in the plane containing the velocity vector and the acceleration vector. The speed and the angular velocity of $\vec{e}_t$ and $\vec{e}_n$ in their plane remain constant. The angle θ becomes

$$\theta = \frac{B}{v}t. \tag{5.25}$$

Equations (5.17), (5.18) reduce to

$$x_p = \frac{v^2}{B}sin\theta, \tag{5.26}$$

$$y_p = \frac{v^2}{B}(1 - cos\theta) \tag{5.27}$$

which describe a circle with its center at $x_p = 0$, $y_p = v^2/a_n$.

The breaking of the initial acceleration into a tangential component A and a normal component B is ill defined when the velocity vanishes. However, the velocity that immediately builds up is in the direction of the acceleration. For this reason we interpret all the acceleration in this case as being tangential *

$$A = a, \qquad B = 0. \tag{5.28}$$

VI THE PHUGOID EXTRAPOLATION SCHEME

Both of the schemes of the previous section ignore some of the basic facts of life of ordinary flying: that the acceleration of gravity $\vec{g}$ is significant compared to specific forces created by an airplane; that a tangential component of $\vec{g}$ will cause significant changes in speed; that speed is a necessary ingredient in generating normal loads. All of these observations are reflected in the phugoid extrapolation scheme introduced in this section.

We assume that the four degrees of freedom at the disposal of the pilot are used to maintain:

1. Constant angle of attack.
2. Constant thrust.
3. Constant bank (mode a) or zero rate of roll (mode b).
4. Coordinated flight (zero sideslip angle).

Assumption 1 translates into both lift and drag being proportional to v^2 with constant coefficients. Using the decomposition of $\vec{a}$ and $\vec{g}$ into tangential and normal components (equations (3.8), (3.9) and (4.2) through (4.4)) we postulate

$$\vec{l} = -\vec{k}Pv^2, \tag{6.1}$$

$$a_t - g_t = Q - Rv^2, \tag{6.2}$$

where P, Q, R are constant. Note that P is positive by the definition of k (equation (4.5); R is positive because of the nature of drag; Q is positive so long as the thrust is in the direction of flight.

The bank of the velocity frame is the angle between the z axis and $\vec{g}_n$. This angle is determined from the initial $\vec{l}$ and $\vec{g}_n$ and, in mode (a), maintained constant throughout *.

The constant P can be determined from the initial values of the variables in equation (6.1). However, equation (6.2) is insufficient to determine Q and R individually. To overcome this a value of aerodynamic efficiency

$$\eta = \frac{\text{Lift}}{\text{Drag}} = \frac{P}{R} \tag{6.3}$$

must be obtained or assumed. Armed with a value of η, equation (6.2) may be restated as

$$a_t - g_t = Q - \frac{P}{\eta}v^2. \tag{6.4}$$

* More rigorously, decompose the acceleration at a slightly later time δt into longitudinal and transverse components, and take the limit $\delta t \rightarrow 0$. By a well known theorem of calculus about the limit of a quotient with both numerator and denominator vanishing, $lim\, A = lim\, \frac{\vec{a}\cdot\vec{v}}{v} = lim\, \frac{\vec{a}\cdot\vec{a}}{a} = a$. Similarly $lim\, B = lim\sqrt{a^2 - A^2} = 0$.

* However, when going through the vertical instant of an inside or outside loop, continuity requires that bank be reversed between right-side-up and upside-down.

With P determined by the initial value of (6.1), Q may be obtained from the initial values in (6.4). With these constants in hand, we can now determine the trajectory for all time.

The governing differential equations are:

$$\dot{\vec{r}} = v\vec{e}_t, \tag{6.5}$$

$$\dot{v} = Q - \frac{P}{\eta}v^2 + g_t, \tag{6.6}$$

$$\dot{\vec{e}_t} = -Pv\vec{k} + \frac{\vec{g}_n}{v}. \tag{6.7}$$

We also have

$$\dot{\vec{e}_t} = \vec{\omega} \times \vec{e}_t. \tag{6.8}$$

Now express $\vec{\omega}$ and $\vec{g}$ in component form as

$$\vec{\omega} = \omega_1\vec{i} + \omega_2\vec{j} + \omega_3\vec{k}, \tag{6.9}$$

$$\vec{g} = g_1\vec{i} + g_2\vec{j} + g_3\vec{k}, \tag{6.10}$$

We equate (6.7) to (6.8), and substitute the appropriate components of (6.9) and (6.10), to obtain

$$-\omega_2\vec{k} + \omega_3\vec{j} = -P\,v\,\vec{k} + \frac{g_2\vec{j} + g_3\vec{k}}{v}, \tag{6.11}$$

which yields

$$\omega_2 = P\,v - \frac{g_3}{v}, \tag{6.12}$$

$$\omega_3 = \frac{g_2}{v}. \tag{6.13}$$

Maintaining these values for ω_2 and ω_3 enforces the condition of coordinated flight. However, the roll rate, ω_1, is still unspecified.

In mode (a), the roll rate is determined by the condition of constant bank. Maintaining the angle between $\vec{k}$ and g_n, requires that the ratio of the components of $\vec{g}_n$ (g_2 and g_3) must remain constant. This is expressed as

$$\frac{d}{dt}\left(\frac{g_2}{g_3}\right) = \frac{\dot{g}_2}{g_3} - \frac{g_2\dot{g}_3}{g_3^2} = 0 \tag{6.14}$$

From

$$\dot{\vec{g}} = \vec{\omega} \times \vec{g}, \tag{6.15}$$

we obtain

$$\dot{g}_2 = \omega_1 g_3 - \omega_3 g_1, \tag{6.16}$$

$$\dot{g}_3 = \omega_2 g_1 - \omega_1 g_2. \tag{6.17}$$

Substituting equations (6.16) and (6.17) into equation (6.14) and solving for the roll rate, we find

$$\omega_1 = g_1\frac{\omega_2 g_2 + \omega_3 g_3}{g_2^2 + g_3^2} \qquad \text{(mode a).} \tag{6.18}$$

In mode (b) , equation (6.18) is replaced by

$$\omega_1 = 0 \qquad \text{(mode b).} \tag{6.19}$$

With the above equations for the rotational rates and accelerations, the position of the airplane and its velocity system orientation are defined for all time. They can be computed by numerical integration.

The selection of the mode used for determining the roll rate, mode (a) or mode (b), is based on the attitude. So long as both pitch and bank stay within predetermined thresholds, mode (a) is used. Should one of the thresholds be exceeded, mode (b) is selected. Mode (a) stabilizes pull-ups and pushovers into gentle climbing or descending spirals or straight flight paths. This is representative of ordinary flying and transport type maneuvers. Mode (b) is suitable for acrobatic flying, and will easily describe vertical and oblique loops. Mode (a), if maintained with the pitch attitude approaching the vertical, results in rapid rolling. The rate of roll diverges as the pitch attitude becomes 90° up or 90° down. However, long before a condition like this is reached, we will have switched to mode (b).

It must be stressed that the phugoid scheme extrapolates velocity orientation whereas the DIS entity PDU conveys body orientation. An attempt to substitute body orientation for velocity orientation in the extrapolation algorithm may cause significant errors. It is, therefore, necessary to extract the velocity orientation before the phugoid extrapolation scheme is applied. This may be done by the method of Section III. In the case of the phugoid scheme, it also makes sense to correct velocity orientation to body orientation for display purposes, since the angle of attack is kept constant. Because of this, the orientation correction described in section III is incorporated into the phugoid scheme.

The phugoid algorithm is expected to predict approximately correct motion so long as the pilot maintains coordinated flight and has not changed the thrust, the angle of attack, and the bank, or the rate of roll (as appropriate for the mode in force). New information will be required only upon new initiative from the pilot. Should communications be momentarily disrupted, the extrapolated airplane would maintain its bank or rate of roll, and, through mild phugoid motion, would hunt and find the angle of climb or descent that the current combination of angle of attack and thrust mandate.

The phugoid scheme requires an estimate of η, the aerodynamic efficiency. The accuracy of this parameter is not crucial. It determines the damping of the phugoid motion. The frequency is independent of it. It is probably permissible to set it arbitrarily at a reasonable value, e.g., 10. In the alternative, unused space in the appearance PDU could be employed to convey this information when the phugoid scheme is being used. The value could then dynamically change as the airplane configuration changes with deployment of flaps or gear or the dispensing of external stores. In this way, maximum accuracy could be achieved.

VII EXCEPTIONS

The equations presented above for extraction of the orientation and for extrapolation of the trajectory are valid for most flight conditions. However, there are a few discrete conditions for which the equations fail. These conditions, and the methods that were employed to work around the singularities they cause, are described in this section.

The procedure for extracting the orientation from the trajectory (Section III) becomes undefined under two conditions:

1. The method aligns the x_v axis with the velocity vector. This cannot be done when the aircraft translational velocity is zero. For this case, we fall back on the method of constant angular velocity, and determine the new orientation from the previous orientation, based on the rotational rate and the elapsed time.
2. The method also fails when the active normal specific force, $\vec{l}$, vanishes (flight at "zero g"). In this situation, the bank cannot be determined. Instead, the aircraft is assumed to achieve its new orientation by the smallest rotation that aligns the x_v axis correctly. This is a rotation about an axis that is perpendicular to both the original x_v axis and the new x_v axis.

The phugoid scheme depends on resolving quantities such as the acceleration of the airplane and the acceleration of gravity into longitudinal and transverse components. This cannot be done when the velocity vanishes. For this reason, when the velocity is below a predetermined limit, we fall back to either of the orientation extrapolation schemes described above and to trajectory extrapolation using constant earth acceleration.

Note that a traditional airplane cannot achieve zero speed except momentarily at the point of a tail slide. At that point the airplane undergoes a violent change of orientation. No harm would result if the phugoid approximation adopted a "do nothing" rule in the case of vanishing velocity, so long as it is applied to an airplane. The reason for falling back on the more traditional approach at extremely low speed is to try to accommodate other vehicles, e.g., helicopters.

VIII EVALUATION

The algorithms of the previous sections were coded as C subroutines and embedded in a program that exercised them for evaluation against recorded flight histories. We used a record of F-16 simulated flight (Table 1) and and a UH1 helicopter flight simulated at our own Lab (Table 2).

The F-16 maneuvered vigorously in the vertical plane, going through two consecutive loops. The angle of attack was high throughout, exceeding 20°. This made the angle of attack correction essential. With the correction in effect both orientation extraction and the phugoid did very well.

Inspection of Table 1 shows that the use of extraction with correction (EC) for predicting orientation, more than halved the number of required updates as compared with constant angular rates (CAR), the method of extrapolating the trajectory being the same. What is more, the results employing EC are very close to comparison data using the true orientation. In other words, coupled with either constant earth acceleration or CLTA, EC is nearly perfect.

The table shows only modest improvement when CLTA is substituted for constant earth acceleration (CEA), with the method of predicting orientation remaining the same. The preexisting combination of constant earth acceleration and constant angular rates (CEA+CAR) was doing pretty well in requiring only 81 updates for the 1409 frame history. The all new combination of CLTA+ EC reduced this further to only 30 (a factor of 2.7). The phugoid scheme did even slightly better, requiring only 27 updates (a factor of 3.0).

Table 2 presents similar data for the helicopter history. This data was recorded from the UA FDL UH-1 simulator flown by Capt. Mark Jackson of the US Army. The file was edited to eliminate a phase including hover and backward drift, which the airplane prediction methods were not designed to handle. The tabulated results show no significant difference between constant earth acceleration and CLTA. Extraction with correction (EC) still does significantly better than constant angular rate (by a factor of 1.8). The all new combination of CLTA + EC (75 updates for 3338 frames) outperforms the preexisting CEA + CAR (137 updates). The phugoid is again the best, but, at 73 updates, only slightly better than CLTA + EC.

The first line of table 2 exhibits a curious effect where orientation prediction by constant angular rates is hurt by improvement in trajectory prediction. Using the true trajectory does worst of all. This point needs further investigation, as do the specific aspects of helicopter prediction.

The benefit of CLTA alone appears to be only modest. The benefit of the assumption of coordinated flight is quite dramatic. The angle of attack correction is essential for the success of EC and of the phugoid. The phugoid is only slightly superior to the combination of CLTA and EC. One tends to conjecture that the phugoid method derives most of its benefit from having coordinated flight built in.

Table 1: Performance Comparison Using F-16 Data

Number of updates in 1409 frames to maintain a tolerance of 10 *ft* and 10°.

	CEA	CLTA	Phugoid	True Traj.
CAR	81	78	NA	75
EC	36	30	NA	22
Phugoid	NA	NA	27	NA
True Orient.	33	28	NA	0

Table 2: Performance Comparison Using UH-1 Helicopter Data

Number of updates in 3338 frames to maintain a tolerance of 10 *ft* and 10°.

	CEA	CLTA	Phugoid	True Traj.
CAR	137	139	NA	160
EC	76	75	NA	13
Phugoid	NA	NA	73	NA
True Orient.	60	58	NA	0

In conclusion, the new extrapolation techniques offered here were able consistently to better the preexisting method by a factor ranging between 1.8 and 3.

ACKNOWLEDGEMENTS

The work described here was done under Subcontract SO-242825-A to Loral Western Development Labs, and first reported in reference 3. We are indebted to Dr. Kurt Lin of UCF IST for providing a 35 second history of F-16 flight. We would like to thank Cpat. Mark Jackson of the US Army for piloting the simulated helicopter flight analyzed in the paper.

REFERENCES

1. S. Goel and K. Morris, "Dead Reckoning for Aircraft in Distributed Interactive Simulation", AIAA Flight Simulation Technologies Conference, Hilton Head, August 1992, p 277.
2. IEEE Std 1278-1993, "IEEE Standard for Information Technology – Protocols for Distributed Interactive Simulation Applications", Institute of Electrical and Electronics Engineers, New York, May 12, 1993.
3. A. Katz and K. Graham, "Extrapolation of Airplane Flight", UA FDL report 93S03 prepared for Loral Western Development Labs, San Jose, CA under Subcontract SO-242825-A, item 04, December 17, 1993.
4. A. Katz "Notes on Dead Reckoning in the DIS Standard", Position Paper, Sixth Workshop on Distributive Interactive Simulation, Orlando, March 1992, vol. II, p 115.

AIAA-94-3429-CP

THE GENERIC SIMULATION EXECUTIVE AT MANNED FLIGHT SIMULATOR

James Nichols*
Naval Air Warfare Center Aircraft Division
Patuxent River, Maryland

Abstract

The Manned Flight Simulator (MFS) at the Naval Air Warfare Center Aircraft Division (formerly the Naval Air Test Center) was created to provide rapid response to a wide range of US Navy simulation requirements. The necessity to simulate any aircraft in the US Navy inventory stimulated the idea of creating "roll-in, roll-out" simulation bays that would accept any cockpit having standard geometric and electrical interfaces. The capability to use any cockpit at any simulation bay in turn led to the need for a flexible and generic software package for simulating any airframe. The Controls Analysis and Simulation Test Loop Environment (CASTLE) executive allows the user to easily generate and operate an aircraft simulation, while also providing a very powerful set of tools for simulation development and engineering analysis. Although the CASTLE package was originally designed to operate on Digital Equipment Corporation (DEC) machines using the VMS operating system and DEC screen management software, recent developments include a MOTIF-based window interface environment and compatibility with the UNIX operating system. The CASTLE package is being proposed as a starting point for a standard airframe simulation package to satisfy general aircraft simulation requirements.

Introduction

Traditionally, the simulation community has been plagued with widely varying requirements that result in numerous simulation architectures. Although meeting the specifications for one project, these architectures usually cannot be reused without considerable time and effort. Some facilities have one simulation package for the pilot-in-the-loop task, and completely different environments for "desk-top" engineering analysis and other tasks. In some cases a significant effort may be required to take a simulation of the same airframe from one environment and re-host it in another. One common example of this is taking an engineering analysis simulation and moving it to the piloted cockpit facility. Although many cases like this may stem from computing platform incompatibilities, the lack of standard simulation software plays a major role as well.

Another long-standing practice has been to take an existing airframe model, copy it, and modify it as required to generate a simulation of another aircraft. This is certainly a valid approach, but results in having to maintain several sets of code that do essentially the same thing. The time and money constraints that commonly accompany simulation projects dictate that a swift and efficient method be used to develop a new simulation without the software maintenance nightmares.

* Aerospace Engineer, Member AIAA

The CASTLE package resolves both of these issues by providing an environment that already contains the "shell" of modules necessary to simulate a simple linear model of an airship up to a highly non-linear model of the most complex aircraft under development.

The CASTLE "Shell"

The CASTLE package consists of several parts. These are the CASTLE "shell" modules, the simulation development tools, and the airframe-specific modules (provided by the simulation developer). The shell modules include the user interface routines, the variable accessing tools, the engineering analysis packages, the airframe executive routines, and the external communications routines. Figure 1 illustrates the hierarchy of the CASTLE components.

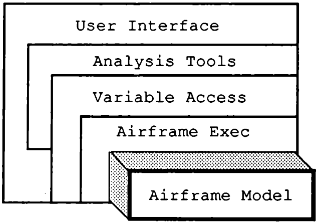


Figure 1: CASTLE Component Hierarchy

The FORTRAN language is used for the airframe executive and is the expected language for the airframe-specific modules. The airframe model itself may be programmed in another language, such as Ada, but interface packets must be created to communicate between the FORTRAN equivalence statements of the CASTLE routines and the other software environment.

User Interface Routines

The user interface routines are intended to provide a user-friendly graphical environment for controlling a CASTLE simulation. The routines communicate to the body of the CASTLE code through several shared memory blocks, so that other user interface packages may be "plugged in" instead of the standard MOTIF windows environment. This method also allows the user interface package to run as a separate task from the airframe executive if desired, and allows commands to come from external sources such as command files. The ability to use command files to submit CASTLE images to background jobs is desired for computationally intensive engineering analysis tasks. There are currently two forms of the user interface, one using the DEC screen management system, and the other using the MOTIF window environment. Figure 2 shows the communication paths between the user interface routines and the CASTLE utilities.

The user interface environment is designed in such a way that adding new facilities is relatively simple. The user must supply a set of routines that define the new screen format, fill the screen buffer as required, and take data from the screen buffer and load it back into the CASTLE tables. The new facility command is added to a command table and the required "action" routines are created for the CASTLE airframe executive. The use of standard templates greatly simplifies this task.

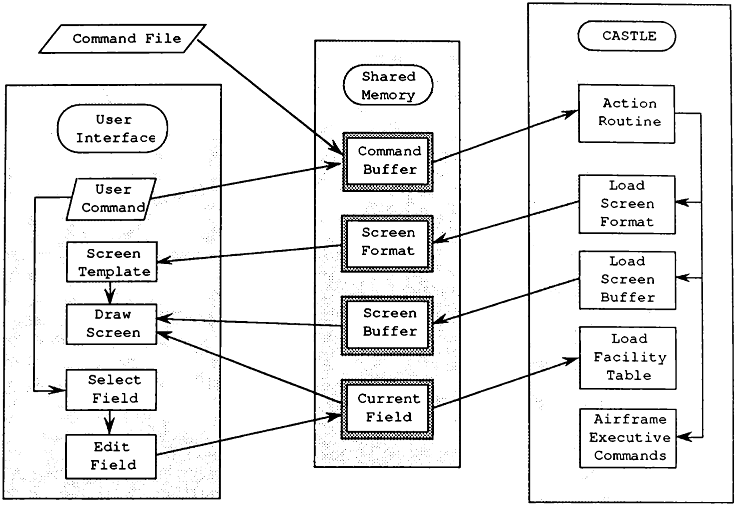


Figure 2: User Interface Connection to CASTLE

Variable Accessing

The cornerstone of the CASTLE code is the method of accessing variables during a simulation session. Earlier versions of CASTLE performed the variable access by searching the symbol table created by the VAX VMS debugger for the specified variable name. The variable address was then found, and the variable value was retrieved from the address location based on variable type. This technique has been abandoned in favor of a machine-transportable method.

As before, the user specifies a variable to be acted upon by entering an ASCII string or selecting the variable from a list. The resultant ASCII string is matched to an entry in a file that contains common block definitions for previously documented variables. The location within the common block is noted, and the address of the variable is calculated based on the common block address, the base address of the variable within the common block, and any array offsets requested.

This symbolic manipulation of variables during program execution provides for some extremely powerful simulation capabilities. It gives both the user and the CASTLE run-time facilities almost unlimited access to any variable that has been properly documented in a Common Descriptor File (CDF).

Common Descriptor File (CDF)

A Common Descriptor File (CDF) contains the documentation for all variables that will define a common block. Each common block that will potentially be accessed by CASTLE must have a corresponding CDF. Each record of a CDF contains the required information about a specific variable located in the "parent" common block. Table 1 describes the CDF record format used by CASTLE. The Common Descriptor Files are built and modified by using the EDITCOM utility and the CASCOMP pre-compiler.

Record Index	INTEGER
Variable Name	CHAR*32
Array Element Name	CHAR*32
Array Specification (as FORTRAN would use)	CHAR*40
Data Type Code	INTEGER
Byte Offset (from start of common block)	INTEGER
Variable Description	CHAR*40
Variable Class (Aero, Engine,etc.)	CHAR*20
Units	CHAR*20
Sign Convention	CHAR*20
Date Modified	CHAR*14
Author	CHAR*20
Alignment Filler	CHAR*6

Table 1: Common Descriptor File Record Format

Common Descriptor File Editor (EDITCOM)

The EDITCOM utility allows the user to interactively build and modify a Common Descriptor File. The CDF is stored on disk in a binary file, and thus cannot be accessed directly by a standard editor. EDITCOM ensures that the record structure integrity is maintained, and performs bookkeeping tasks such as byte offset calculation and allocation of space for arrays.

CASTLE Pre-Compiler (CASCOMP)

The CASCOMP pre-compiler tool operates on FORTRAN modules with a standard CASTLE header and performs two basic tasks. The first task is to automatically generate FORTRAN "EQUIVALENCE" statements for all variables that are described as either "inputs" or "outputs" in the glossary section of the file header. The CASCOMP routine searches all selected Common Descriptor Files and locates these "global" variables. The variable data type as declared in the FORTRAN source file is compared to the true definition stored in the applicable CDF. If the data types do not match, a warning is issued. The EQUIVALENCE statement is generated based on the name of the common block described by the appropriate CDF, and the byte offset retrieved from the CDF. An array element may be described by a unique name other then the actual "root" array name. An example of this would be an integer array "ROOT(20)" located in the common block XBLOCK at a byte offset of 5 that has a tenth element named "TENTH". The resulting EQUIVALENCE statement generated by CASCOMP would be:

```
EQUIVALENCE(XPAR(45),ROOT(10),TENTH)
```

where "XPAR" is a byte array mapped to the XBLOCK common block. The common block declarations are also generated automatically, and the user may elect to have CASCOMP generate the FORTRAN data type declarations as well.

The second task performed by the CASCOMP pre-compiler is to build a CDF for each FORTRAN module for "local" variables. A "local" variable is defined as a variable that is not included in an existing "global" CDF, but may still be accessed during a simulation session. The local variables may be requested by adding them to the "LOCALS" section of the glossary definition in the file header. The MAKELCL utility may also be used to build the "LOCALS" section, and will request all declared variables that are not listed in the INPUTS and OUTPUTS section of the glossary.

Analysis Tools

The primary engineering analysis tools that are built into the CASTLE package are the Maneuver Function Generator (MANGEN), the trimming facility, the Linear Model Extraction (LME) utility, and the Simulation Checking using an Optimal Prediction Evaluation (SCOPE) package.

Digital time history data is stored by data sets that point to locations in a large data buffer. As many as 50 different time histories may be stored simultaneously for comparison or use by the analysis tools. Figure 3 describes the data storage structure used internally in CASTLE. The digital data may be saved to external files in a variety of standard formats, as well as formats defined by the user. The digital data may also be stored externally during a pilot-in-the-loop session by passing it through shared memory to a real-time output routine, allowing virtually unlimited storage capacity.

Maneuver Function Generator

The Maneuver Function Generator (MANGEN) allows the user to drive any available variable with a pre-defined function or combination of functions. MANGEN will also accept digital time histories as driving inputs. It is mainly intended to serve as a substitute for real-time pilot control inputs. Another utility under development is a preliminary version of a virtual pilot, which will be used as a true aircraft maneuver generator. This will allow the user to define a target flight condition and have the aircraft fly to the specified condition and execute the requested maneuver. It could also be used for commanding maneuvers such as "S-turns", wind-up turns, and level accelerations.

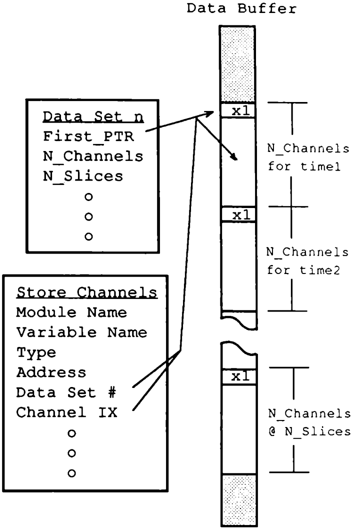


Figure 3: CASTLE Time History Storage

CASTLE Trim Facility

The CASTLE trim facility perturbs the user-defined trim controls to drive the user-specified state derivatives to target values for a requested set of initial conditions. The trim routine may use any available variable for a control or state derivative, and thus is extremely versatile. It may be used to trim out asymmetric store loadings, trim to a flight path, or trim in any steady state flight condition. Some pre-defined conditions that may be selected include constant-rate turns, pull-ups, and push-overs.

Linear Model Extraction

One of the most useful utilities in CASTLE is the Linear Model Extraction (LME) facility. It allows the user to define a linear model structure using ASCII simulation variable name strings for states, inputs, state derivatives, and outputs. The A, B, C, and D matrices that pertain to the equation set:

$$\dot{\mathbf{x}} = [A]\mathbf{x} + [B]\mathbf{u}$$
$$\mathbf{y} = [C]\mathbf{x} + [D]\mathbf{u}$$

where:

$\dot{\mathbf{x}}$: state derivative vector
$\mathbf{x}$: state vector
$\mathbf{u}$: input vector (controls,etc)
$\mathbf{y}$: output vector

are determined by perturbing the states and inputs from a pre-defined state and measuring the effect on the state derivatives and the outputs. Several perturbations are used with varying step sizes to get the best approximation of the partial derivative for each matrix element. The output results may be written to a MATLAB® "M" file for further analysis. The LME utility can be used to generate a linear model of any portion of the airframe model by using the pre-defined model selection types or by the judicious use of the airframe executive module enabling flags.

Simulation Checking Using an Optimal Prediction Evaluation (SCOPE)

One of the more frustrating tasks in developing an airframe simulation is getting it to perform like the actual article. The SCOPE package allows the user to drive the simulation with time history data and make a qualitative and quantitative comparison of the simulation output to the test criteria time history. It can be used to get an estimate of the initial conditions biases as well as average biases on the total forces and moments that will minimize the error between the input and output data. The initial condition biases help prevent the simulation from immediately diverging from the test criteria data. The biases on the forces and moments identify which portions of the model may need attention. The Low Order Estimation Tool (LOST) and LOST eXtension (LOSTX) utilities are used to identify, build, and test corrections to the model without recompiling code. The LOSTX utility may be used to incorporate these correction factors without changing the baseline model.

Airframe Executive

The airframe executive consists of the airframe loop executive, the equations of motion, the atmosphere model, ground interface routines, and templates for the airframe model routines. The equations of motion are a highly modified version of the SMART routine used in the NASA Ames BASIC simulation package[1]. The atmosphere model uses the ARDC62 standard model and incorporates the ability to define nonstandard day ambient temperature and pressure, as well as density and pressure altitudes. Steady state winds aloft are added to random turbulence and step gusts if desired. A burble model is available for the carrier landing environment, and a turbulence grid generated from Computational Fluid Dynamics (CFD) software is being incorporated for LHA-class ships.

Airframe-Specific Requirements

The airframe developer is responsible for providing airframe-specific modules to satisfy the airframe loop executive. Table 2 lists the procedures that the loop executive will call. If a user-specific routine is not supplied, a dummy routine will be inserted instead. The airframe modules may be coded in any language, as long as the variables are accessible in a common block and documented properly.

COCKPIT	Cockpit controls/switches
IMPORT	External inputs (piloted)
ELEC_SYS	Electrical systems
HYD_SYS	Hydraulic systems
SENSOR	Air data, gyros, etc.
CONTROL	Control laws
SURFACE	Surface actuators
ENGINE	Propulsion systems
AERO	Aerodynamics
WAITIN	Weight, cg and inertias
MISC_FM	Miscellaneous (sling load)
EXPORT	External outputs (piloted)

Table 2: Airframe-Specific Modules

The airframe programmer is free to replace any CASTLE module with a customized version if desired, but the new version must reside outside the baseline CASTLE libraries. If enough interest is generated, the new version may be incorporated in the production CASTLE package. The airframe executive looping routine is sometimes replaced due to specific airframe requirements.

Function Table Processor (FTP)

Most simulations use some form of function table lookup for the massive data that accompanies an airframe model. The CASTLE environment encompasses a Function Table Processor tool[2] that takes function data and converts it into FORTRAN functions. The form of the function access is very easy to understand when trying to interpret the top-level code, and looks like:

```
CLALFA = CLALFA_FTP( MACH, ALT )
```

where MACH and ALT are the independent arguments.

AEROPLOT

The AEROPLOT utility is a separate subset of the CASTLE package. It is primarily used to drive the complete aerodynamic model by sweeping selected parameters and observing the output with plot graphics or digital analysis. This is most useful during the initial development phases, as it checks the end-to-end aerodynamic data and catches improperly implemented data functions and equations. It is also good for depicting the aerodynamic coefficients in different axis systems. The effect of a control surface or state on the total aerodynamic model may be observed also. A capability to use AEROPLOT as an engine for generating coefficients not explicitly included in the model is being incorporated as well. The AEROPLOT package uses the standard CASTLE user interfaces and variable accessing tools. The airframe programmer must supply an interface routine between the AEROPLOT executive and the same airframe code that is linked to the CASTLE simulation. Although AEROPLOT was originally developed as an aerodynamic modeling tool, any portion of the airframe simulation may be linked to AEROPLOT. It is an invaluable method for doing end-to-end checks of a subsystem model.

The Near Future

The CASTLE package is continually adding new capabilities in response to the ubiquitous "what if...?" asked by the users. The modular interfaces and coding designs make such expansions a generally straightforward process. One such effort is underway to tightly couple CASTLE and sophisticated analysis tools such as MATLAB® and other packages that perform Parameter Identification (PID) and similar advanced techniques.

Conclusion

The CASTLE simulation environment developed at Manned Flight Simulator represents a considerable effort to satisfy as many simulation requirements as possible while retaining as much modularity and flexibility as possible. As such, CASTLE makes an ideal candidate for a standard simulation package.

References

[1] McFarland, R.E., A Standard Kinematic Model For Flight Simulation at NASA-Ames," NASA CR-2497, January 1975.

[2] Nichols, J.H., "MFSFTP User Guide Version 3.2," Naval Air Warfare Center Aircraft Division, Maryland, 1994.

AIAA-94-3430-CP

A PROCESS FOR THE DEVELOPMENT OF SIMULATION STANDARDS

Bruce L. Hildreth
SAIC/Systems Control Technology Group
100 Exploration, Suite 2005
Lexington Park, MD 20653
301-863-5077

Introduction

The proliferation of flight simulations throughout the United States and the world is not a surprise to any informed aerospace industry observer. The extremely high cost of aircraft operation combined with the fact that much training that a pilot needs cannot be performed safely in an aircraft. One such example is the training of emergency procedures. Additionally, in the past several years the incorporation of weapons systems into simulations has increased the use and importance of simulation. Obviously, you can simulate combat in a much more cost effective manner than using real weapons.

However, to many industry observers, it is a surprise that there has been very little standardization throughout the simulation industry. It is apparent that there are many areas which standardization could take place so this lack of standardization is somewhat baffling. The fundamental physics behind aircraft, missile and space vehicle flight are the same regardless of how an aircraft looks or its size. The information that is required to model an airframe and its environment is the same regardless of the airframe. The particular values that information takes may be different for different aircraft but the type of data that is required is fundamentally the same.

This paper presents a logical and straightforward process in an attempt to define what areas can be standardized most easily and furthermore, to define a logical process of creating standards in the area of aircraft simulation. This process could also be used as a guideline in the development of standards in other areas of simulation.

This paper does not attempt to recommend simulation standards. Several standards based on the process discussed in this paper will be recommended in upcoming papers.

Existing Efforts Involving Simulation Standards

Several significant efforts are presently involved in the area of simulation standards. Some of these efforts are listed here for reference.

The Defense Modeling Simulation Office (DMSO) has a major effort in simulation networking standards. This is called Distributed Interactive Simulation (DIS).

DMSO also sponsors the Modeling and Simulation Industry Working Group which is pursuing several efforts in simulation standards.

The Advanced Project Research Agency (ARPA) is funding the Joint Modeling and Simulation System (JMASS), a major effort which includes simulation modeling. ARPA also funds the Software Technology for Adaptable Software Reuse (STARS) program which includes the development of some reusable simulation software.

The United States Air Force (USAF), the International Air Transport Association (IATA), and the Federal Aviation Administration (FAA) are all working on standards for simulation certification and testing.

The USAF sponsored Project 2851 has developed standards for visual system data bases in simulation.

The USAF is presently updating their simulator development guideline document AFGS-

Division Manager, SAIC
Member AIAA

87241 Rev. A, "USAF Guide Specification, Simulators, Flight", Jan. 1989, to incorporate significant software architecture changes.

As part of the Computer-aided Acquisition and Logistics System (CALS) program, the standard DOD 8320.1-M-1, "Data Element Standardization Procedures", Jan. 1993, has been issued. It specifies database format standards applicable to simulation.

The National Aeronautics and Space Administration, as part of the High-Speed Research Phase II program, is attempting to develop standards for the exchange of simulation models between participating agencies and contractors.

The Software Engineering Institute at Carnegie Mellon University has developed software design paradigms for flight simulators.

Last, but not least, the AIAA Flight Simulation Technical Committee has assumed the goal of promoting simulation standards in the US Government and industry. Their initial goal is to standardize the export /import of mathematical models between simulation users (Step 1 discussed below).

Criteria for the Development of Standards

The process presented is intended to satisfy the criteria presented below. The development of simulation standards should:

1. Minimize short term impact on the user;
2. Maximize long term application of the standards;
3. Maximize long term benefit to the users;
4. Result in reusable software;
5. Result in life cycle cost savings;
6. Include testing and validation;
7. Include the flexibility needed to adjust to changing user requirements;
8. Include documentation requirements;
9. The standards developed should be as language independent as possible.

To meet these criteria, the following process characteristics are required:

a) The process should be evolutionary in nature;

b) The process should allow incorporation of other standards;

c) The process should be based on a building block approach starting with simple standards with restricted scope and proceeding to more detailed standards with a wider scope;

d) The more detailed standards (higher level standards by our definition) should incorporate the simpler standards (lower level standards, again by our definition);

e) The process should include life-cycle support of both the standards developed and all reusable software developed;

f) The process should not be dependent upon software language used or compilers used. (Note: the standard may be dependent upon a particular language but the process used to develop the standard should not be.)

Based on the above requirements, the process discussed in the remainder of this paper was developed. It is based on the principle of defining the interface between simulation facilities as the starting point and building upon that interface definition. Each standard that would be produced more rigorously defines that interface until each simulation user (or facility) is using some of the same software modules and some of the same database definitions (with different data in some cases).

We must remember that, by definition, standards apply only to what is "standard" or "common" between two or more users. Therefore, in the process for creating and applying standards, we must recognize that no standard can be applied to all users all the time.

The Proposed Process for Developing Simulation Standards

This section will describe what simulation sub-systems could most easily be standardized. There are seven different general areas for which standardization is immediately required and may be initiated in a straightforward manner. These are listed below, generally in order of ease of standardization.

1 Standard Data Interchange Format
2 Standard Common Variable Definitions
3 Standard Common Database Definitions
4 Standard Testing and Validation Methods
5 Axis System Definitions
6 Standardization of Module Functionality
7 Standard Source Code

The task of standardization is actually easier than it might first appear. This is true not only because all dynamic aircraft simulations are based on the same laws of physics, but also because the standardization can be performed in an evolutionary manner. The standardization can progress from the simple (standard variable names and meaning) to the complex (standard reusable high level code). Each level builds upon the standards developed at the lower levels. This fact dramatically eases the pain which standardization would otherwise cause. But the process will still not be painless. This evolutionary standardization process is illustrated in Figure 1.

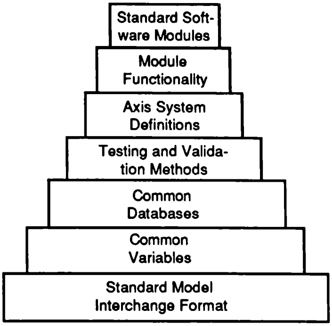

Standard Software Modules
Module Functionality
Axis System Definitions
Testing and Validation Methods
Common Databases
Common Variables
Standard Model Interchange Format

Figure 1. The Recommended Evolution of Software Standards Applied to Flight Simulation

Process Step 1: Define and Standardize the Interface Between Users

Just as the foundation of a home is key to the structural integrity of the home, this step is key to the integrity of the entire standards development process. It is key not only to the quality of the standards, but also to the acceptability of the standards by the user community.

The first logical step is to define and standardize the simulation modeling information sent between users. This forms the foundation for all other standards developed. They will all build upon this standard. It is also key to user acceptance of the standards because this standard is of immediate benefit to them, allowing them to more easily import simulations from other users.

This level of standard essentially provides the user with standard import/export rules. The user need not change anything inside his facility or software. The only change required is to write a utility to input/output data to/from his internal format to the data exchange format. This is, in general, a simple task and, in fact, several candidate standards exist at this level.

This level of standard is foundational in that, to exchange information between users at least the following must be clearly defined;

a) The variables or parameters being exchanged. This includes clear, unambiguous description, units, sign convention, axis system (if applicable), etc.;

b) Data format for the exchange. This includes parameters, time history data, and function table data; and

c) Method of verifying the information was exchanged correctly.

The above have clear relationships to and set the foundation for later levels of standardization including common variables, databases, testing, and axis systems. However, at this level of standard, they are easier to define because they are more limited in scope and, therefore, easier for the user to accept.

Process Step 2: Common Variable Standardization

All simulations have many variables that are common between simulations. Examples of these in airplane simulations are angle-of-attack, Mach number, altitude, true air speed, angle-of-sideslip, etc., and with definition of axis systems standardized, lift coefficient, drag coefficient, sideforce coefficient, pitching moment coefficient, yawing moment coefficient, rolling moment coefficient, pitch rate, roll rate, yaw rate, etc. There is no reason to have angle-of-attack be a different variable name and definition in each simulation. In addition, the physical definition of "true angle-of-attack" between simulations is often actually different. In some simulations, "true angle-of-attack" does not include the change in angle-of-attack due to turbulence, while in others it does. This gives tremendous potential for mistakes in software modification and obviously has significant software life cycle cost implications. The standard must name and define common used variables. The definition must include name, an unambiguous description, units, sign convention and axis system used (if applicable).

Process Step 3: Common Database Definitions

As in the case of the common variable standard, the standardization of database format should be easy to implement.

The airframe model databases, while different for each aircraft, should have a common format. Airframe databases include environment, aerodynamic, engine, and flight control function tables which define the non-linear characteristics of the environment and airframe. It is very simple to standardize the format in which function table data is specified so that different airframe models could be exchanged between facilities. Standardizing the functional database format of airframe models is an important step both in transferring models between facilities and developing standard airframe models. This standardization should include whether the data is orthogonal (one set of independent arguments for each independent variable) or non-orthogonal (a separate set of independent arguments for each dependent variable). Standard table look-up and interpolation software could then be easily developed later.

This standard should include configuration control and documentation requirements. Modeling databases include thousands of data points which are frequently modified from their original values. The source of the data and any modification made should be documented.

Standard Simulation Databases

Databases that can be standardized include:

navigational aids;
threats databases;
ship and airfield databases;
visual system databases and;
aerodynamic/airframe model databases.

Most immediately, both the navigation aids database and the airframe model databases can be standardized at a very low cost and with very low industry impact. The navigation aids database could be literally identical for every trainer that simulates flight in the U.S. and in foreign countries. Almost every simulator has the same database requirements for navigation aids such as TACAN, OMEGA, ADF, ILS and global positioning systems.

Like the navigational aids, the requirements for ships and air fields are in general, identical between simulations. The database for ships and air fields could readily be standardized.

Process Step 4: Testing and Validation

The purpose of this level of standard is to simplify the process of determining that a simulation is properly installed on a particular computer system, not to verify that the simulator flys like the airplane. In addition to standardizing software and data in simulations, there exists a need for standard test capabilities in simulators. This is due to the fact that if it is desirable to have standard software or databases that are used in more than one facility, the capability to test the implementation in a consistent manner is also desirable. Just as in databases and software, there are certain test capabilities that are required for any aircraft simulator. These common test capabilities are discussed below.

At least four important testing capabilities are needed as a standard minimum. These are:

a) the capability to store time history data from a simulation test run;

b) the capability to display the data recorded;

c) the capability to insert standard test inputs; and

d) the capability to look at variables and insert values interactively (peek and poke).

The capability to record simulation variables is very common on simulators. Practically speaking, the standard for this requirement will need to focus more on what data must be stored and the user interface so that the testing process between facilities becomes standard. The method and format of archiving this data to disc or tape should also be standardized so that the archived data can be exchanged between users. Indeed, this is expected to ultimately become one of the key methods for importing and exporting test data between facilities.

The need for the display of the data is obvious. The display capabilities must allow both the plotting of variables as a function of time and cross plots of several dimensions. Again, most simulators have the capability to do this; however, the capability varies and the user interface and procedures vary creating tremendous confusion of what data is actually presented. This standard data output capability is required to standardize the validation and testing of simulators.

The key to the validation testing even of simple systems in the simulators is the ability to easily input simple and known test inputs. While the ultimate acceptance of any simulator is the acceptability of the simulator to the pilot (or operator), prior to operator acceptance testing the system should be tested with known computer reproducible inputs. A simple example of this is to test the longitudinal response of an aircraft math model at a given flight condition, a simple computer generated step our doublet input may be used. Obviously, more than one input is needed to validate the implementation of a model, but they all can be generated by software.

Finally, the simulation should have the capability to, as the simulation is running, examine variables on a terminal and change those variables. This capability is commonly called peek and poke. This would be used to debug and also as part of the test procedures when the exact value of a variable is needed. The peek and poke facility must also have a method of hard copy.

Key to the testing and validation step in standardization is the documentation requirements. Previously standardized data formats and variable naming can be used to define the data that must be plotted to verify the plotted data against baseline documentation. Once this documentation for a particular simulator is produced, it can also be used for configuration control and recursion testing.

Process Step 5: Axis System Definitions

As part of the standardization of the variable names and aerodynamic functional data (that has been discussed above), the standardization of axis systems will help refine those standards and is a prerequisite in standardizing the equations of motion used in the simulation. Without standard axis systems, the simulations will have different equations of motion and different aerodynamic functional data in the very least.

A major constraint on the axis system definition is the requirement to insure compatibility with the aircraft avionics that may be integrated into the simulation. In tactical aircraft, for example, the avionics system includes a model of the world that may be a rotating round or oblate spheroid earth model. Therefore, in order for the equations of motion to coincide with the weapons system calculations similar earth models are required. For example, the F-18 AYK-14 mission computer uses a geocentric (or round) earth model. However, for longer range weapons systems and as computer capabilities increase, the weapons systems may require oblate spheroid or even higher fidelity earth models. However, a standard axis system can be used for any of these models.

The axis systems considered for standards must be considered as a set and include such axis systems as;

1) Stability Axis
2) Body Axis
3) Velocity Axis System
4) Earth Fixed Axis.
5) Geocentric Inertial Axis System
6) Orbit defined Axis System
7) Heliocentric Axis System

The American Institute of Aeronautics and Astronautics (AIAA) and American National Standards Institute (ANSI) have jointly published "Atmospheric and Space Flight Vehicle Coordinate Systems" (AIAA/ANSI R-004-1992). This is a candidate standard for flight simulation to use for axis systems.

Process Step 6: Software Modeling Standardization

After axis system standardization, the next step would be to standardize the functional requirements of certain modules or program packages. For example, certain modules can be well defined with regards to what function they perform and certain functions are needed for any airframe simulation. The objective of this level standard is then to make it clear what modules should perform each function in order to ease software maintenance and to further ease transfer of software and data from one user to another. At this level of standardization, converting software from one system to another becomes a matter of "re-wiring" (assigning of inputs and outputs) particular modules in the simulation rather than requiring the re-design of several modules. As long as the functional definition of each modules remain the same, the variable names may change, the method of performing the calculations may change, but the interface with the rest of the simulation does not change. If the object/class definitions, function prototypes, header file, and/or COMMON variable, etc. standard has already been adopted then the "re-wiring" requirement is dramatically simplified.

For example, the aero simulation modules should be standardized to require that the inputs to the aero module are all true values of output parameters of the simulation. For example, the inputs to the aero module should be true Mach number, true angle-of-attack, true temperature, true airspeed, true altitude, etc. Variables such as indicated angle-of-attack, indicated airspeed and so forth would not be inputs to the aerodynamic simulation. The outputs for the aerodynamic modules can be rigorously defined as forces and moments in the body axis system referenced to the airframe moment reference center. With this convention, as long as simulations adhere to these relatively straightforward and simple constraints, it would be much easier to move an aerodynamic simulation of an airframe from one simulation site to another.

This convention also applies perfectly well to engine, flight control, and landing gear simulations. Similar conventions can be arrived at for the simulation of electrical, hydraulic, and fuel systems.

Other functional definitions which are simple to standardize include atmosphere, wind turbulence, equations of motion and axis transformations. The atmosphere, equations of motion, and axis transformations are particularly important so that there can be consistency between the response of one simulator vs. another. Different software versions of the equations of motion result in a slightly different dynamic response. In general, the difference in response is negligible, but it can significantly increase the difficulty in comparing or testing one simulation vs. another. For example, if an airframe simulation is moved from one site to another and there is a slight difference between simulations at the two sites, if they have different equations of motion you are not sure whether to attribute these differences to the equations of motion or to the incorrect implementation of the airframe model. This issue becomes extremely important in validation testing.

Process Step 7: Standard Software Modules: Reusable Software

The final level of standardization

would be to use the exact same code throughout the industry for certain modules. This software should adhere to all the standards discussed above.

The modules that are most readily standardized are listed below in order of priority.

Function Table Processing Software
Equations of Motion
Atmosphere
Axis Transformations
Wind and Turbulence Model
Navigational Aid Models
Ship (aircraft carrier, LHX, etc.) and Airfield Models
Test and Validation Software (as described above)
Record and Playback Software
Weapons Software
Threat Software (both ground and airborne threats)

Program Management and Life Cycle Support Requirements

Any standard proposed will cause much anguish in the simulation industry. Everyone will complain basically because their system was not chosen as the standard. There is no solution to this problem and the organization that champions the standard must be prepared for this.

There must be an organization to maintain the standard specifications and standard software and disseminate copies to the users requiring them. Life cycle support of both software and standards is often neglected. Life cycle support of the standards and reusable software is a critical part of both.

Life cycle support is required to update the standards, software or databases, fix bugs and add and delete capabilities as required. In addition the support organization should be required to identify additional standards or software for standardization and would have a group of personnel responsible for software testing. This support could be started with a very modest staff. Since standardization should be performed in an evolutionary manner, there would only initially be a very few reusable functions or modules. The number of reusable modules would then grow with time.

Another task this support must perform is liaison with the simulator procurement and life cycle support organizations to identify which procurements should be required to use the standards and/or reusable software modules. Standardization should be required on most procurements but there are always exceptions to the rule that cause the standard reusable software to not be able to satisfy some system requirements. In many cases where the standard reusable software cannot meet the requirements of the simulation, either the procurement specification is too demanding or the standard software can easily be upgraded to meet the requirement of that particular procurement.

Figure 2 illustrates the possible functional organization for the required simulation standard life cycle support.

This organization shall be very easy to create since it can be a matrixed group of offices at different locations. In fact, having different organizations make up the standardization office will probably be the optimum total organization in that the experts in each technical area (such as threats, airframes, visual systems, navigational aids, etc.) would probably be the best organization to create and maintain the standard software in that technical area. These offices could be formed from (or the task responsibility added to) existing organizations. With the low cost / high speed data communications available today these offices could be located throughout the country and be linked via network. Each facility would be responsible for its own reusable software and would have standard configuration controls and procedures. The simulation standardization program office would coordinate all activities and perform the high level software design. This organization is conceptually shown in Figure 3.

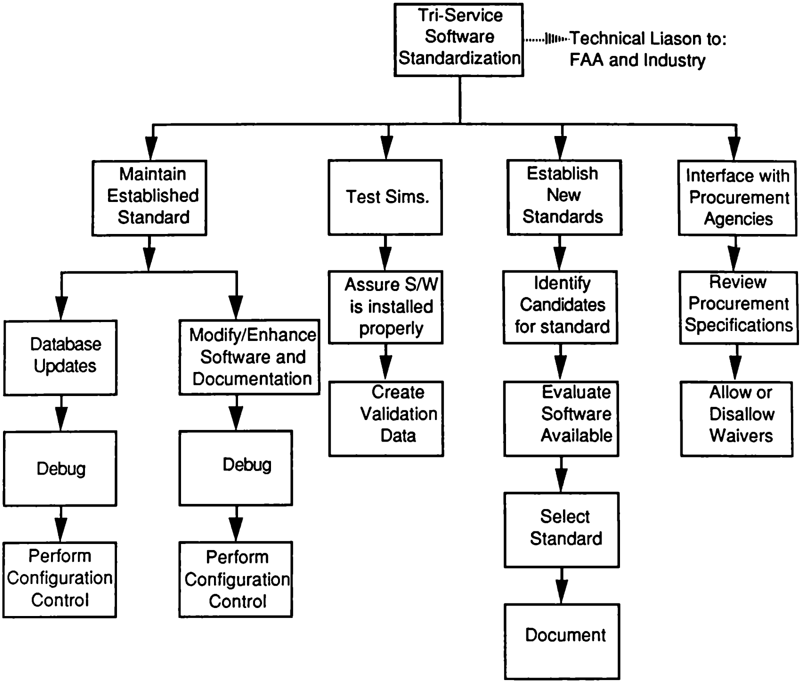

Tri-Service Software Standardization
Technical Liason to: FAA and Industry
Maintain Established Standard
Test Sims.
Establish New Standards
Interface with Procurement Agencies
Database Updates
Modify/Enhance Software and Documentation
Assure S/W is installed properly
Identify Candidates for standard
Review Procurement Specifications
Debug
Debug
Create Validation Data
Evaluate Software Available
Allow or Disallow Waivers
Perform Configuration Control
Perform Configuration Control
Select Standard
Document

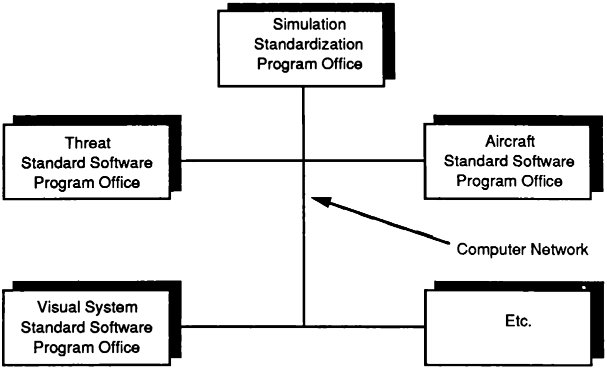
Figure 2. Function Responsibilities and Possible Organization of the Simulation Standards Office

Conclusions

Standardization in simulation is easy to perform if done in an evolutionary manner, adopting the simplest standards first and building on these simple standards one step at a time until there is ultimately standards and related reusable software modules.

This paper has presented a straightforward process for developing simulation standards and reusable software. The process starts with a standard for the export/import of simulations between users and facilities. The process continues with increasing levels of standards: standard simulation variables; standard database formats; standard testing and validation; standard axis system definitions; standard module functionality; and ultimately resulting in standard, reusable software. There is then the requirement for the life cycle support of both the standards and the reusable software.

Standardization at any level will reduce overall program acquisition and maintenance cost without loss in performance. In addition, as the level of standardization increases, the financial benefits gained will likewise increase, especially as standard software modules and databases are supplied as part of a procurement, rather than being uniquely regenerated for each system and requiring software support unique to that software or database.


Simulation Standardization Program Office
Threat Standard Software Program Office
Aircraft Standard Software Program Office
Computer Network
Visual System Standard Software Program Office
Etc.

Each Office Has Standard Software Development and Configuration Management Tools and Procedures.

Std Sim Prog Org

Figure 3. Standards Control Organization Can Have Several Offices at Different Sites Specializing in Software Covering Their Respective Technical Expertise

AIAA-94-3431-CP

DEVELOPMENT OF A DEPLOYABLE AH-1W TRAINER

Chad C. Miller
David Perdue
SA103 / SY30
Flight Test and Engineering Group
Naval Air Warfare Center
Aircraft Division
Patuxent River, Maryland 20670
U.S.A.

ABSTRACT

The U.S. Navy's Naval Air Warfare Center - Aircraft Division has designed and produced a mobile AH-1W Supercobra Aircrew Procedures Trainer prototype using Commercial Off-The-Shelf (COTS) products. The prototype is deployable by air, land or sea and is equipped with the Distributed Interactive Simulation (DIS) gateway for networking. The program demonstrates that COTS technology can, for a low price, produce high fidelity portable training systems that can re-use even very complex existing software packages. This paper presents a summary of the design and construction process that resulted in a successful blending of existing software and new technology hardware, and comes to conclusions that are applicable to many existing training needs.

LIST OF ABBREVIATIONS

DIS	Distributed Interactive Simulation
COTS	Commercial Off-The-Shelf
NAWC/AD	U.S. Navy's Naval Air Warfare Center - Aircraft Division
MFS	Manned Flight Simulator Facility
WST	Weapons System Trainer
OFT	Operational Flight Trainer
IOS	Instructor-Operator Station

PROBLEM STATEMENT

The reality of the "New World Order" for most military organizations following the transformation of the Soviet Union is an era of declining resources and funding as the perceived threats decline.

In military aviation this often translates into more and more reliance on simulation equipment for training and a movement away from the use of actual flight hardware and flight hours.

Modern technology has produced the "Weapons Systems Trainer" or the "Operational Flight Trainer" in the U.S. Department Of Defense as an answer to the added requirements placed upon simulation in training. OFTs are being used as a substitute for actual aircraft flight instruction for basic flight training. This is an expansion of the role of simulation in military training beyond the use primarily for emergency procedures training that was common only a few years ago. WSTs are used for basic flight training, but are also designed to do the additional functions required to train the student, or re-fresh the veteran, in delivering accurately the large amounts of ordnance carried by today's attack aircraft. WSTs have supplanted the actual live firing of modern, expensive, weapons to the extent that many pilots who have not flown in combat may have never fired one of the more exotic weapons that they are expected to be able to use in combat, or at best are allowed to fire one or two such weapons a year.

Modern WSTs and OFTs do a good job of training modern pilots for the threats and scenarios that they may meet in a still-hostile world, but they share one flaw that has become more apparent to air crews and training systems personnel alike: a fixed site availability. Current WSTs and OFTs are large, building sized devices, often requiring dozens of support staff and technical crew members to ensure daily availability.

One of the lessons learned from the recent Gulf War was that the use of WSTs and OFTs for large amounts of ordnance training assumes that conflicts will rapidly develop and forces would be committed directly from their training areas to combat. Months of waiting for the order to engage were not planned, and forces waiting in the desert were neither able to return to the squadron's fixed site WSTs nor expend ordnance for refresher training purposes. The need for a high fidelity deployable procedures trainer for squadrons deployed away from fixed site WSTs was recognized.

DESIGN REQUIREMENTS

In November of 1992 the Naval Air Systems Command, Code PMA-2052L tasked the Manned Flight Simulator Facility (MFS) of NAWC/AD to develop designs for two deployable AH-1W Marine Supercobra aircrew procedure prototype trainers (APTs). There were to be two prototypes designed, one to meet the training needs of the active duty Fleet Marine Force and one tailored to the needs of the Reserve squadrons. The major differences between the two devices were the second device would have a functioning optical sight unit for use in weapons delivery procedures training. The base requirements for the systems were set such that the devices would:

- use existing software whenever possible
- use COTS equipment whenever practical
- be usable within the existing MFS facility
- be deployable by air, land or sea
- be capable of operation within one day of deployment to a new location
- be capable of operation by a single pilot without leaving the crewstation area
- be capable of operation using portable power supplies
- be capable of non visual, or visual system based training.
- use actual aircraft equipment or current WST parts whenever practical
- incorporate all current fleet modifications to the aircraft, including the new Tactical Navigation System.
- be capable of training all procedures, both normal and emergency, found in the NATOPS.
- be required to perform limited weapons scoring functions
- be night vision equipment compatible

In addition, the prototypes were to be ready for deployment testing no later than 30 months after project start.

Once construction of the prototype devices is complete, the devices are to be deployed to training command sites for use and evaluation by the training squadrons. After use by the training commands, comments and suggested design changes will be made to the technical specifications of the device. A production decision for follow-on devices will then be considered.

METHODOLOGY

The MFS facility is a high technology Navy organization composed jointly of the Strike Aircraft Test Directorate and the Systems Engineering Test Directorate of NAWC/AD. MFS was selected for the prototype construction because the unique nature of the facility and the type of simulation work done there could translate directly into the AH-1W APT project.

The MFS facility has long been involved with modular simulation systems, and supports over 10 high fidelity aircraft simulation programs with a 40 foot dome and a six degree of freedom motion base station as the primary work stations with the use of modular, mobile cockpits. These cockpits are configured to be able to move from one cockpit station to any of the other three cockpit stations and become fully operational within a half hour. The projects are as diverse as the V-22, F/A-18E/F and the X-31. MFS therefore had experience in building cockpits that were essentially self-contained. The modular software structure in use at the facility could also re-use, with some modification required, the existing AH-1W WST tactical, aerodynamic, control, propulsion and other related software.

A key element in the MFS design of simulation software architecture is the use of a common memory area, shared among all the laboratories' machines. This device, called the multiport memory, allows any computer system on the cluster, whether it be a host processor, visual image generator, or a specialized piece of hardware, to share information at 800 kilobytes per second. The use of such a shared memory directly benefits the Cobra APT, by allowing the MFS software to work as designed, and to allow the DIS network equipment access to the information required.

The approach taken by the design team was to use existing assets of the Navy whenever possible, to minimize cost and unique items in the design. This design ideal was followed even when more attractive, but higher cost and more complex, solutions to design problems were found. Examples of areas where more costly solutions could have provided increased fidelity are the control loading system and the cockpit canopies structural design.

BASIC DESIGN OVERVIEW

The design team quickly decided to base the AH-1W APT prototype on existing aircraft crewstation fuselage sections, which are available in large numbers. The fuselage section containing the crew stations, or "cockpit" was to house all equipment required to operate the fully functional APT with the exception of the power conditioning equipment and the visual image generation system.

The system was designed to fit within a mobile unit complex consisting of three standard mobile units with minor modifications. These units house the deployed system. In addition, these three units serve as the shipping containers when the device is being transported. The device design called for a three channel visual system for the first device, and a four channel visual system for the second device. The final system design layout is show in Figure 1.

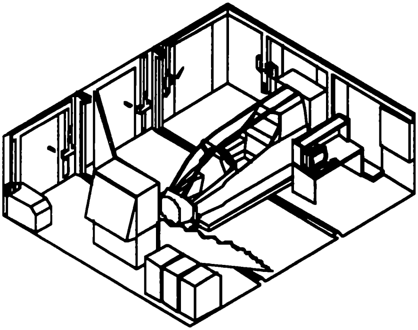


FIGURE 1
System Design Layout

COMPUTER ARCHITECTURE

The computer system chosen for the APT was the Digital Equipment Corporation's VaxStation Model 4000-90, with the VMS operating system. The MFS has had extensive experience with this type of system in past years, and the modular software systems developed there operate best with this environment. The DEC Model 90 has a Linpack Single Precision rating of 12.91 million floating operations per second (MFLOPS). This rating compares to the current host machines of the AH-1W WST's 1.8 MFLOP rating. To allow large amounts of space for future expansion in input/output requirements, it was decided that two Model 90s with BIT3 Corporation's Turbochannel to VME adapters would be used as the host machines for each APT. A shared VME backplane serves as the common Input/Output system for the APT and as the path to the shared memory. One of the BIT3 VME adapters contains a daughter card populated with 2 megabytes of shared memory. The majority of accesses are performed by the airframe VAX, so its BIT3 VME card hosts the shared memory. This minimizes VME bus utilization. A FORCE 68040 VME Single Board Computer (SBC) is used to process DIS protocol data units and is also located in the VME chassis. The SBC DIS software utilizes one megabyte of the shared memory to communicate with the visual system software.

One machine was dedicated to the airframe model and visual system communications, and the other machine would perform the avionics and instructor/operator functions. The airframe directly accesses the stick and collective analog to digital converter to minimize transport delay. In addition, the machines were equipped with Military Standard 1553 Bus interfaces for communication with actual aircraft hardware. The basic host system architecture is shown in Figure 2.

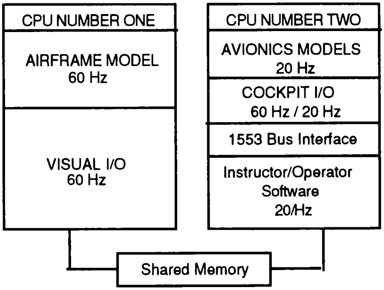


FIGURE 2
Host Computer Software Architecture

The Model 90s measure 40x20x60 Centimeters and weigh less that 75 kilograms each and mount in standard computer racks located at the rear of the cockpit. This can be compared to the room full of computers required to run the same software for the existing WSTs.

Full system testing revealed that the computer system operated with a 30 percent minimum reserve of computational power on each machine that gives the system growth potential as the airframe and avionics models become more complex in the future. The Fleet Project Team pilots greatly preferred a 60 Hz visual image to a 30 Hz visual image, although not due to transport delay. Their main objection to the 30 Hz visual image was the jitter that is visible at high velocity and low altitude. A 75 millisecond average transport delay is achieved with the visual system running the 60 Hz Yuma database and the airframe running at 60 Hz. A simulation with less extensive input/output requirements, such as an Instrument Flight Trainer or Part Task Trainer would perform very well on a single machine. Complete usage of each system is given in Table 1.0.

CPU NUMBER 1	PERCENTAGE
Cockpit Input/Output	11%
Avionics Models	31%
Weapons Models	17%
IOS	10%
TOTAL USED CPU 1	*69%*
CPU NUMBER 2	PERCENTAGE
Stick Interface	1%
AH-1W Airframe	37%
Visual with DIS	33%
TOTAL USED CPU 2	*71%*

TABLE 1.0
Host Computer System Usage

MECHANICAL DESIGN

The APT is based on the forward fuselage section of an existing Cobra aircraft. For the two prototypes one fuselage section was a "T" model aircraft, and one section was taken from an aircraft that had been modified to a "T" to "W" model conversion demonstrator. In both cases the modifications required were similar.

The cockpit sections are mounted on a lifting device equipped base frame to provide stability and structural strength. The cockpit and frame are designed to meet a 2.5 load factor with a 50% safety margin. The base frame is equipped with castoring wheels and hard points for mounting and transportation. The device is equipped with standard computer mounting racks on the aft end, and existing fuel and transmission housing spaces are used also for equipment housing.

For cooling the crewstations, the windscreen material of the cockpit was removed, leaving only a three inch section of the material around the canopy supports for structural strength. This configuration removed the requirement to rebuild completely the canopy support rails of the cockpit section, saving considerable cost and time, but left the cockpit with an altered visual perspective from the design eye points. The cockpit is shown in Figure 3.

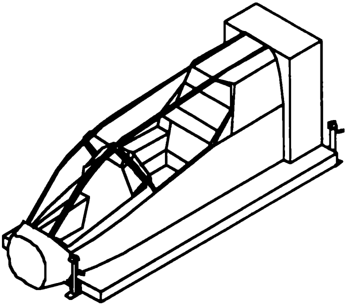


FIGURE 3
Cobra APT Crewstation

CONTROL LOADING SYSTEM

The MFS uses digital electric control loading equipment for reproducing the force-feel systems of the aircraft for the aircrew. Due to the complexity and cost of such systems, the first Cobra APT was designed without such a control loading system. It was decided instead to use the existing aircraft control linkages and force-feel cartridges for pilot feedback. The only additions to the system found in the aircraft would be upgraded position sensors for measuring stick displacements and the addition of magnetic particle brakes into the system to provide a method of sharply increasing the force gradients on the system to replicate a hydraulic system failure condition.

The design team allowed the possibility of installing electronic control loaders into the second APT if the design was inadequate in the first prototype. One of the design aircraft force per displacement curves is shown plotted against the APT's same force per displacement curve in Figure 4, and a dynamic response is given in Figure 5.

Based on this information, and with pilot comments, it was decided that the simple design system would be used for all the systems in the project.

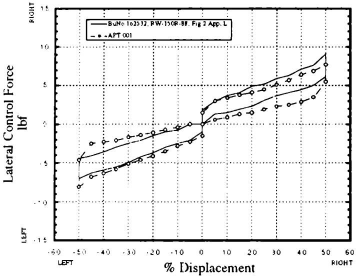


FIGURE 4
Actual vs. Simulated Static Forces

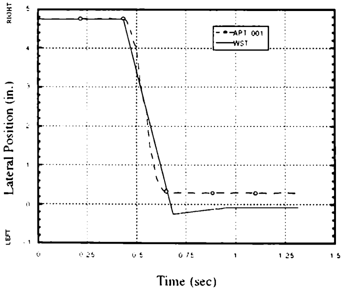


FIGURE 5
Actual vs. Simulated Dynamic responses

VISUAL SYSTEM

The requirements for the APT visual image generation systems were:

- small physical size
- textured visual scenes
- ability to use existing WST databases
- low cost per channel
- able to produce night vision device imagery.

The system selected was the Evans and Sutherland ESIG-2000 image generation system, which fulfilled all the above requirements. In addition, the system is capable of 60 Hz operation for minimal transport delay, laser range finding, and up to 63 moving models. This system is capable of using areas of the visual databases developed for the WSTs.

For the visual display, the design is constrained by the interior height of the standard mobile units. Floor to ceiling height is only 2.0 meters at the lowest points. This effectively ruled out many types of COTS visual system display devices. Three types of display presentations were designed and presented to a group of training command pilots. These configurations used either large rear-projection cabinet type displays or a combination of this type display with large front projection screens. For all the configurations, a single rear projected display was used directly in front of the cockpit. This display was configured as shown in figure 6.

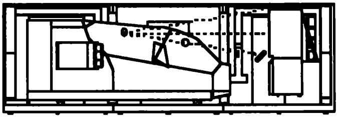


FIGURE 6
Forward Display Field of View

This front display provides an 18.4° vertical field of view, with 9.4° of view up from the horizon and 9.0° field of view below the horizon, with a 24.3° horizontal field of view. Various other configurations were tested to achieve the most training value through the placement of the other two display "windows."

In all cases the display devices were required to be usable with night vision equipment. There were two types of displays considered. One was a large floor-standing rear-projected screen device (show in profile in Figure 6). The other was a low-gain front projected screen, illuminated with high quality projectors mounted vertically on the sides of the cockpit section.

The display configurations are shown in Figures 7 - 9.

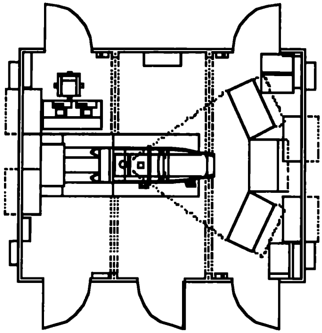


FIGURE 7
Configuration I

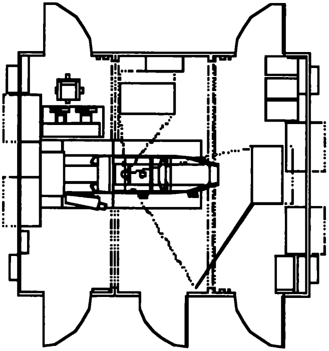


FIGURE 8
Configuration II

The three different configurations are described below:

I: Two rear projected screens flanking the center screen.

II: One rear projected screen located 90 degrees to the left of the pilot position, and one front projected screen to the right of the cockpit.

III: Two front projected displays positioned to the left and right of the center display.

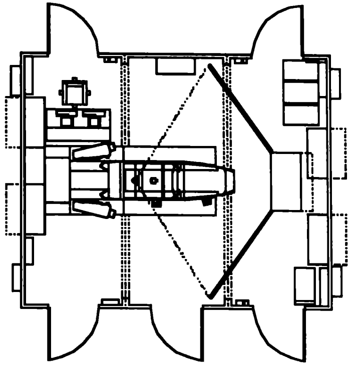


FIGURE 9
Configuration III

The configurations were evaluated during flight conditions ranging from low in-ground-effect hovering flight to rapid flight profiles including rapid stops and radical turns. Finally, the display configurations were evaluated using night vision equipment. The configurations were rated on a scale of 1 (best) 2 (acceptable) and 3 (unacceptable). The results, and the categories of evaluation are given in Table 2.

Display Type	Field of View	Night Vision	Cue Sensing
I	3	1	1
II	2	3	2
III	1	2	2

TABLE 2
Rating of Display Options

It can be seen that configuration I, even though it gave the best image and clarity during both normal and night operations, was found not to be acceptable due to field of view limitations. The aircrew would loose horizon references during rapid accelerations and stops due to the limited vertical field of view. Configuration II was not acceptable due to the large differences in the distance between the front and left display devices. This required the night vision gear used to be re-focused each time the pilot shifted his vision from the front to the left. Configuration III produced acceptable results for both cue sensing and night vision work, with the best field of view of all the options, and was chosen for the final configuration (Figure 1). This provides side fields of view (FOV) of 8.4° up FOV, 26° down FOV, and 45.7° horizontal FOV. This results in a total of 115.6° horizontal FOV.

ELECTRICAL SYSTEM DESIGN

One of the requirements of the design was to employ as many actual aircraft components as possible, including the actual aircraft computational resources, often referred to as "black boxes". This can be more expensive during the construction phase of such a program, but the maintenance cost over the life of the system is lowered. Personnel do not have to be dedicated to maintaining and upgrading any software emulation required in the simulator. In addition, the current military supply system serves as a repair and replacement system, as opposed to being forced to create a supply system for simulator unique devices.

In the APT the actual Heads-Up-Display (HUD), the HUD display generator, weapons system control devices, radar warning indicators and interface control units were used, including the displays and keypads for the new Tactical Navigation System. The HUD required modification to allow the optics to focus on the center display screen, but otherwise all the equipment is unmodified.

For the majority of the static pressure and oil pressure driven instruments of the pilot and weapons operators consoles, it was possible to use the same equipment as was designed and constructed for the WST. This allowed, again, the use of equipment for which a repair and supply system already existed. The only custom created circuit cards needed in the entire device are to stimulate the Radar Warning Receiver Display and the Tacan input of the new Tactical Navigation System.

The entire system operates on one 100 ampere 120/208 Volt, 3 phase 60 Hz external power supply. The mobile unit complex cooling and heating equipment requires two additional such lines. This amount of power is capable of being generated by several U.S. Military standard mobile generators. The APT mobile units contain power conditioning and power interruption protection equipment to allow operation and protection to the system from any quality of power supply.

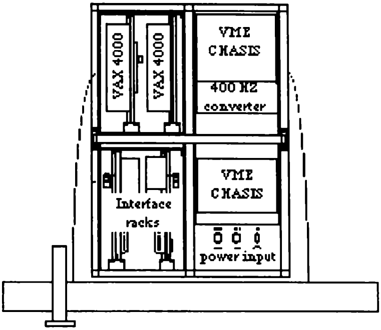


Figure 10
Placement of Equipment in Cockpit
Rear View

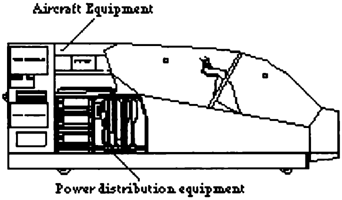


Figure 11
Placement of Equipment in Cockpit
Side View

Figure's 10 and 11 show the placement of all the equipment required to run the fully functional simulation, exclusive of the visual image generation equipment, embedded within the cockpit.

INSTRUCTOR/OPERATOR SYSTEM

One of the requirements for the system was that it would be operable by an instructor with: two pilots, two pilots alone, or a single pilot. This requirement resulted in an X Window menu driven Instructor/Operator System (IOS) design that contains enough flexibility to meet all the training requirements of the APT.

From the IOS menu the user can select several displays to monitor the progress of training. Features include; on-line real-time checklists for the situation encountered that monitor the steps taken and record the mistakes (if any) made, point and select stores for scenario development, computer generated approach maps with real-time position of the aircraft, and a student data log book of APT time used.

Several of the IOS's features exist to allow pilots to use the APT alone or with another aircrew. Malfunctions are selected and set to occur randomly during the duration of the scenario during single user operation. The IOS menus can be accessed from the crewstations without having to exit the cockpit. The crew has a cursor positioning device with buttons mounted just outside the cockpit, within easy reach. Pressing a button freezes the simulation session and places the IOS menus on the large center screen.* The crew can then use the trackball and buttons to manipulate the IOS just as if they were seated in front of it. Exiting the IOS screens resumes the simulation from the frozen state. Many such features allow a pilot to enter the device, select a training scenario, execute the scenario, review his or her performance on the checklists and log the time.

NETWORKING

To make the system as flexible as possible, the design team decided to make the system compatible with the protocols of the DIS network. The DIS network is a system of standard information packages that is passed between systems to allow the systems to interact with one another. The APT is equipped with the necessary equipment to connect to the DIS network.

The system architecture of the APT allowed easy incorporation of the DIS system. The common data area contained all the information required by the DIS packets, and the Model 90 host computers are capable of producing the packets for the interface hardware with ease. The DIS capabilities of the system were demonstrated during the Interservice/Interagency Training and Equipment Conference at Orlando, Florida, USA in early December 1993 and again in May 1994 at Quantico Virginia.

SYSTEM TESTING AND RESULTS

The APT has been undergoing systems testing at the MFS facility since September 1993, and is scheduled for delivery for fleet training squadron evaluation in June 1994. Before delivery, the system will have undergone a series of exhaustive testing, to prove that in every case of similar training scenarios the behavior of the device matches that of the WST. The testing document is based on the exhaustive testing procedures document that was prepared for the acceptance testing of the WST. The APT is expected to match the behavior of the WST simulation systems in all areas that have commonalty. A sample time-history test case is given in Figure 12. This figure shows the response of the APT to a step input compared to the response generated by the WST to the same input. This type of testing ensures that the device will act as a complement to the existing training systems and will not provide negative training to fleet users.

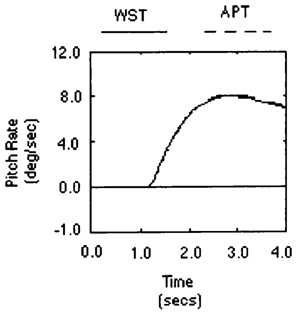


Figure 12
Time History Matching
WST Vs APT

FUTURE PLANS

The Cobra APT will be on-site at Camp Pendleton, USA from June 1994 to February 1995. The second device, which incorporates a fourth channel of visual and weapons scoring,

* U.S. Patent Pending

will be deployed in September of 1995 at a new Reserve squadron site that has yet to be determined. Once user comments and reliability and maintainability data have been gathered, the production of multiple copies of the prototype will be accomplished through open competition using the extensive documentation created by the design and build team.

Incorporation of a full blade element model is underway at the MFS facility to increase the fidelity of the aerodynamic models. Planned equipment upgrades to the aircraft are being designed for incorporation into the APT devices so that the APTs will continue to match fleet configurations.

The Night Targeting System (NTS) will be incorporated into the second prototype. This will place greater processing requirements on the visual and computational components of the APT. The capability of the ESIG-2000 to produce acceptable FLIR displays, the Digital Equipment Corporation Alpha RISC technology, and a host based DIS software package written in Ada will be evaluated.

In addition, the upgrades designed for the APT's will serve as a prototype design for the inclusion of the fleet upgrades into the WSTs. This is possible due to the similarities of the systems in hardware and software. The integration design of the new Tactical Navigation System hardware into the APT has already been delivered to the training command for incorporation into the WST. In this fashion, upgrades to the large fleet trainers can be accomplished in shorter amounts of time, given that the system will already have been "debugged" on the smaller devices.

CONCLUSIONS

-New super microcomputers have allowed modern training devices to divorce themselves from the traditional large cold room with banks of computers traditionally associated with large training devices. Even when a training device requires interfaces with actual aircraft equipment or a system of other networked devices, the total system can be made very small. With the exception of the shipping and shelters for the APT, the crewstations themselves are the largest portion of the device.

-Software from existing training devices can be re-used at low cost to spawn small, portable devices. The only system used for large WST and OFT type trainers that is not currently available within the cost and size constraints are motion systems, which are not used for many fixed wing applications.

-Large future savings can be generated for new training systems if pre-construction consideration is given to re-use of software and hardware between fixed site and mobile trainers.

-The new generation of low cost, small footprint visual systems can meet almost any training and physical space requirements and still provide good cueing and training features.

-The use of a common shared data pool in software systems allows access by a wide range of devices, allowing easy upgrades and additions to systems. The Cobra APT uses its shared memory to interact with the DIS network, the visual system, and the other host computers.

-Small APT type systems can be used to prototype upgrades for existing fleet trainers.

-The Cobra APT required no new inventions or major productions of new equipment. Instead, the program designers simply used existing technology, software, and COTS equipment to produce a rugged, easily maintained and replicated device to create a useful, user friendly and mobile training device.

ACKNOWLEDGMENTS

The author would like to note that the project described in the paper above is the work of the Cobra APT design and construction team at the MFS facility, which has again demonstrated that its ability to produce quality designs within tight time and schedule constraints are without equal.

The authors would also like to recognize the vision of Mr. William Lee of the Naval Air Systems Commands PMA-205 division, whose leadership has made the program possible.

BIOGRAPHY

Mr. Miller is the section head for the tactical aircraft simulation section (SA103) of the MFS facility, a section that encompasses the Navy's F/A-18, F/A-18E/F, AH-1W and EA-6B simulation efforts. He received his B.S. degree in Aerospace Engineering from North Carolina State University of Raleigh, North Carolina, U.S.A. in 1986. Mr. Miller has been a member of the AIAA since 1984.

Mr. Perdue is the senior Technical Specialist in SY30 for the Manned Flight Simulation Facility. He has served in this position as the principal designer of the Manned Flight Simulator since 1986. He received his B.S. degree in Electrical Engineering from Virginia Polytechnic Institute and State University in 1975, and his M.S. degree, also from VPI, in Electrical Engineering in 1976.

AIAA-94-3433-CP

HELICOPTER IN-FLIGHT SIMULATION DEVELOPMENT AND USE IN TEST PILOT TRAINING

R. V. Miller* and L. A. Khinoo**
U. S. Naval Test Pilot School
Naval Air Warfare Center Aircraft Division
Patuxent River, Maryland

Abstract

The U. S. Naval Test Pilot School (USNTPS) trains pilots, flight officers and engineers in the technical and managerial skills necessary to conduct aircraft and airborne systems test and evaluation. An integral part of this training is the use of ground and in-flight simulators to demonstrate the effects of varying aircraft flying qualities parameters. The USNTPS has developed specialized simulation facilities to meet the unique training requirements. This paper describes the development of a Variable Stability and Control (VSC) system installed in an SH-60B helicopter as a specialized training aid for use at the USNTPS. The development of the VSC system is traced from requirements through syllabus introduction.

Introduction

The U. S. Naval Test Pilot School, at NAS Patuxent River, Maryland, is a Directorate of the Flight Test and Engineering Group (FTEG), Naval Air Warfare Center Aircraft Division. (The FTEG was formerly the Naval Air Test Center.) The FTEG is responsible for conducting test and evaluation of aircraft, systems, components, and related equipment. In support of this mission the USNTPS trains test pilots, test flight officers, and test project engineers to conduct these evaluations. Graduates are involved in important decision-making processes regarding future aircraft and airborne systems acquisitions and upgrades.

Overview of Curricula

Instructional Approach

The instruction at the USNTPS is comprised of classroom lectures, briefings, simulation and flight test exercises. The test pilot applicant is a highly motivated and experienced aviator with a B.S. degree and many times an M.S. degree in engineering, mathematics or the physical sciences. The relatively short time span of 11-months necessitates a unique instructional approach. The instruction is aimed at improving analytical flying skills, developing analytical writing ability, and expanding aircraft and mission knowledge through increased exposure to a wide variety of aircraft and systems types. These principles are basic to the course of instruction and have guided the design of the integrated academic, flight and report curricula at the school.

Curricula

The course of instruction offers three separate curricula: fixed wing, rotary wing, and airborne systems. Although common in many respects, each curriculum contains specialized academic courses and flight test exercises.

Academic Instruction

Two to four hours a day are devoted to formal academic instruction with a total of approximately 480 classroom hours. The academic courses are tailored to provide the student with the required technical background to support the flight projects. The academic courses are categorized as engineering, aircraft performance, aircraft flying qualities, airborne systems, and other specialized subjects.

The use of the VSC system is most revelant to the flying qualities courses which address both fixed wing and rotary wing stability and control. In these courses, the basic governing equations of motion are derived and

[*] Academic Instructor
[**] Flight Instructor, Veda Inc.

then used to interpret the various flight test methods. The synergistic effects of control system, cockpit displays, and visual cueing are discussed and demonstrated.

Flight Instruction and Reports

Each student completes 40 to 50 flight projects that include test planning, project flying, and reporting of test results. In each exercise, the technical background developed in the classroom is applied to flight exercises where the student is charged with documenting the characteristics of the aircraft or system in an engineering sense and with evaluating it qualitatively to determine the potential of satisfying operational requirements. On each exercise, engineering data are obtained and processed into an acceptable form for inclusion in the technical report. These data, along with qualitative evaluations pertaining to items of mission relation, are the basis for the final report.

Aircraft

The flight program is supported by a stable of 38 USNTPS aircraft,currently including F-18, T-2, T-38, U-21, U-1, U-6, UH-60, SH-60, H-58, OH-6 and the X-26 glider. During the course, all pilots will fly 12-14 different types of aircraft. In addition, variable stability and control (VSC) aircraft are employed to demonstrate to each student the wide variety of parameters that affect handling qualities. Currently employed VSC aircraft are the Calspan Corporation Learjet Model 24, USAF NT-33 aircraft, and the USNTPS NSH-60B helicopter. The T-33 is soon to be replaced by the Variable stability In-flight Simulator Test Aircraft (VISTA) F-16.

The Use of Simulation at USNTPS

General

The USNTPS has developed specialized ground and airborne simulation facilities to meet the school's unique training requirements. In-flight and ground based simulators are used to teach flying qualities and control system characteristics. A primary objective of the flying qualities instruction is to demonstrate the effects of each of the elements of the pilot-vehicle system illustrated in figure 1. The simulator typically uses a control system loader to vary the desired feel system characteristics such as breakout, friction, and spring gradient. A variable display system is used to present a range of display formats and show the effects of display dynamics.

Use of Ground-Based Simulation

Ground-based simulation has been employed as a training aid at the USNTPS since 1969 [1]. The role of USNTPS ground-based simulation is similar to the in-flight simulation. The advantages of ground based simulation include: more controlled conditions, increased availability, use for demonstration and practice of conditions that would be unsafe for in-flight simulation, lower cost to procure and operate, and an effective way to prototype in-flight VSC exercises.

Unfortunately the results from handling qualities tests conducted in ground simulators are often misleading due to distortions caused by the limited visual and motion systems. One objective of using both ground and inflight simulation is to demonstrate the limits of evaluations conducted in ground simulators. The increased use of ground based simulation in RDT&E requires that the test team understand these limitations and the utility of ground base simulations. The exposure of the USNTPS student to a range of ground and inflight simulators provides the background necessary to fully utilize simulators as a T&E tool.

The use of ground-based simulation represents an important step in the systematic progression from the concepts presented in the classroom to the real world environment where the flight tests are performed. The high fidelity Manned Flight Simulator at the NAVAIRWARCENACDIV and the low cost USNTPS simulators provide the facilities necessary for effective training of the test pilot and flight test engineer.

Because of these relative advantages and limitations, the ground-based simulation role is viewed as being complimentary to in-flight simulation and is being actively pursued along with the in-flight simulation.

In-Flight Simulation Via a VSC System

In-flight simulation has been used as a teaching tool at USNTPS since the introduction of the B-26 VSC aircraft in 1960. The unique role of the VSC aircraft programs has been to provide flying laboratories in which the student evaluator can relate the engineering parameters of aircraft stability and control to their effect on the flying qualities of the aircraft. With the VSC aircraft, the evaluator can be exposed to a wide range of flying qualities in realistic mission related airborne tasks.

In-flight simulators are implemented via a VSC system which is a modification of the flight control system that allows in-flight variation of the flying qualities of the host aircraft. The major elements of the pilot closed loop system are the pilot, display system, feel system, aircraft, sensors, VSC computers, and servo actuators (figure 1).

VSC Operating Modes

VSC operating modes are typically categorized as "response feedback" or "model following". These modes are described below.

Response Feedback VSC - In this mode the VSC system parameters controlled are the feedback, feedfoward, and shaping parameters. The VSC parameters are closely related to aircraft stability derivatives. The aircraft modal characteristics would typically vary with flight conditions.

Model Following - In this mode the VSC system parameters are model parameters such as frequency or damping ratio. The pilot's control input drives a VSC math model and the VSC system then attempts to drive the VSC aircraft to follow the math model response. In this mode the aircraft response characteristics tend to be invariant with aircraft configuration and flight conditions.

The response feedback mode is used in all of the current USNTPS applications.

Use of VSC in Curriculum

A chronology of the significant VSC system milestones at USNTPS is

1960 -	First VSC syllabus flights in B-26
1972 -	VSC NCH-46 introduced
1974 -	VSC NT-33A introduced
1981 -	VSC Learjet model 24 introduced
1981 -	VSC X-22A V/STOL introduced
1982 -	VSC NCH-46 modernization program completed
1987 -	Helo VSC replacement program initiated

The variable stability flight program consists of a series of coordinated classroom lectures, ground-based simulations, preflight briefs and flight demonstrations. The flights are spaced throughout the school year and address topics such as basic aircraft dynamics, test techniques, advanced flight control systems, and handling quality evaluations. The flights build on the concepts introduced in the academic courses and enhance the ability of the prospective test pilot to make handling quality evaluations. In addition, flight experience is expanded by showing the evaluation pilot a variety of unusual and undesirable aircraft characteristics. These flights have proven to be extremely valuable as a tool to teach the fundamentals of analysis and correction of flight control system deficiencies.

During the 11-month program, each pilot participates in five VSC training flights. The engineer and flight officer receive up to three VSC flights. All of the VSC flight and associated lectures, except the VSC NSH-60B, are conducted by Calspan. The annual syllabus use of VSC aircraft at the USNTPS is approximately 200 flights for the VSC Learjet and 45 flights for the NT-33A. The projected annual utilization of the NSH-60B in the VSC training role is approximately 50 flights. A summary of the VSC flights for the Rotary Wing curriculum are summarized below.

Rotary Wing VSC Flights

Flight	Aircraft	Subject
1	Learjet	Long Stab Demo
2	Learjet	Lat-Dir Demo
3	Learjet	S&C Review
4	NT-33A	Advanced Flight Controls Demo
5	NSH-60B	Lat FQ Demo
6	NSH-60B	Response Types & Coupling

NSH-60B VSC Program Overview

Background

A program was initiated in 1987 to replace the CH-46 VSC aircraft ^{2}. A chronology of significant program milestone is

1987 - NCH-46 replacement program initiated
1988 - VSC NCH-46 returned to the fleet
05/91 - VSC requirements established - Statement of Work (SOW)
12/91 - VSC SH-60 contract awarded
06/91 - SH-60B delivered to contractor
10/91 - Ground test
10/91 - Phase I flight test
11/91 - A/C ferried to USNTPS
01/92 - Phase II flight test starts
09/93 - Interim VSC flight exercise
06/94 - VSC I for USNTPS Class 106
08/94 - VSC II for USNTPS Class 106

Engineering and syllabus development tests for a third VSC exercise are scheduled for 1994.

Airframe Selection

The selection of the H-60 series helicopter as the host VSC aircraft was based on aircraft availability, supportability at the USNTPS, and the demonstrated SAS authority and bandwidth. A review of existing flight test data, analysis, and manned simulation test, indicated that the approximate ± 10% SAS authority would provide adequate angular acceleration for a VSC system in the roll and pitch axes, and marginal angular acceleration in the yaw axis. In June of 1992 an SH-60B aircraft BuNo 162974 was transferred to the USNTPS for the VSC system installation.

Test Aircraft Description

The NSH-60B is a single piloted, single main rotor, twin-engine helicopter manufactured by Sikorsky Aircraft. The main rotor system consists of a fully articulated, four-bladed rotor with a hinge offset of 4.7%. The tail rotor, mounted on the starboard side, is a four-bladed, tractor type, canted 20 degrees from the vertical. The side-by-side cockpit has dual conventional flight controls. The hydraulically boosted and irreversible flight controls incorporate mechanical mixing to minimize inherent control coupling. An AFCS is incorporated to assist the pilot in maneuvering and controlling the aircraft. The AFCS is composed of three major subsystems: the Stability Augmentation System (SAS), Stabilator System, and Digital Automatic Flight Control System (DAFCS). The aircraft maximum gross weight is 21,700 pounds. The sonobuoy launcher, spectrum analyzer, and other mission avionics equipment have been removed. A flight test instrumentation system and the VSC system have been installed. The aircraft can be reconfigured by removing the VSC kit so that mission avionics such as the Radar or Doppler TACNAV, and multipurpose display can be re-installed.

Instrumentation

The VSC Helicopter is equipped with a MARS 2000 wide band airborne recorder, a PCM encoder and associated pre-conditioning filters. Up to 48 channels of analog data at 85 samples per second may be recorded with 12 bit resolution.

Helicopter VSC Requirements

The USNTPS VSC basic program requirements were to replace the function of the NCH-46 VSC and, where feasible, upgrade the school's helicopter in-flight simulation capability. An assumption was that the rotary wing student would continue to use the Learjet, T-33, and ground base simulator and that the VSC would be primarily used in the hover.

The VSC system will allow the demonstration of the effects of varying stability and control parameters about the pitch, roll and yaw axes. The response feedback method was specified based on its successful use in the CH-46 and other USNTPS VSC applications. Other significant SOW requirements included:

- Maximum use of host aircraft sensors and actuators
- Notch filters for specified sensor signals
- First order lead/lag in signal flow paths
- A safety trip system (method not specified)
- Ability to change VSC configuration within 30 sec.
- VSC data bus
- Servo command bus
- Selected inter-axis coupling
- VSC in kit form
- Test plan for ground and flight test
- Frequency response test for SAS actuators

VSC Implementation

The design fabrication and installation of a VSC in an SH-60B was contracted in 1991. The host aircraft was delivered to Calspan in June of that year. The Calspan approach [3] for this design was to use an adaptation of the proven Learjet VSC system.

The VSC hardware installed includes:

- VSC Control Panels
- VSC Electronics Enclosures
- SAS/VSC Relay Transfer Enclosure
- 3-Axis Rate Gyro Package
- Position Transducers

There are no special provisions for varying the cockpit control feel characteristics.

The NSH-60B VSC uses the existing ship's SAS actuators and attitude gyros sensors. Additional VSC sensors are control position transducers and VSC rate gyros. The various feedback signals are then shaped (first order lead-lag), amplified, and summed according to the programmed response and sent to the limited authority SAS series actuators. Sensor signals from one channel can be cross coupled to the other channels.

VSC Operation

When the VSC is engaged, inputs to the limited authority SAS actuators are transferred from the host aircraft flight control computers to the VSC computer via the SAS/VSC Relay Transfer.

Typically the safety/instructor pilot occupies the left seat and the evaluator/student occupies the right seat. The VSC can be flown from either seat when the VSC is engaged.

The VSC can be manually disengaged by either pilot if an unsafe condition is encountered. The VSC also has automatic safety trips that disengage the system. The trips are actuated when the difference between the commanded and actual actuator position is greater than a specified threshold (in volts), for a specified time. The specified values for the development program were 4 volts (10 volts is full actuator throw) and 100 ms. When disengaged, control of the SAS actuators revert back to the host aircraft flight control system. The reversion is to either a SAS ON or SAS OFF aircraft depending on which mode was selected at engagement.

The response-feedback flight control system is programmed through digital computer controls located in the cockpit.

Configuration Control System

The configuration control system (CCS) replaces the manually adjusted potentiometers used in the VSC NCH-46. The CCS allows up to 256 VSC configurations (sets of 64 VSC system gains) to be quickly set during flight or ground operation. Of these 256 configurations, 128 are permanent and the remaining 128 temporary configurations can be defined and stored during normal flight or ground operation.

VSC Flight Test Program

Objectives and Scope

Initial program objectives were to safely develop the maximum VSC gain envelope and then develop two syllabus exercises within this envelope. The two exercises would be similar in purpose to the NCH-46 VSC I and VSC II exercises previously used in the curriculum.

Tests were conducted during two phases. Phase I tests, acceptance and functional evaluation, were conducted at the Calspan flight test facility in Buffalo, N. Y. The Phase II tests, engineering and syllabus development, were conducted at NAWCAD, Patuxent River.

Phase I test consisted of ground tests and 3 test flights totaling 6 hours. Phase II of tests were conducted at Patuxent River consisted of 27 test flights totaling 50.6 flight hours.

Phase I Tests

The objective of the Phase I flights was to verify the basic functioning of the system and to make sure that the systems was performing correctly before returning to USNTPS. Specific tests objectives were to:

- Verify airborne function of the system and fulfill contractual requirements for VSC System Acceptance
- Evaluate the aircraft fault monitoring systems with VSC engaged
- Verify safety trip system operation and adjust thresholds
- Assess input signal quality and implement filters
- Generate baseline open loop data
- Initiate the gain increases
- Monitor system performance to identify any tendencies toward high frequency instabilities

Phase II Tests

Follow on Phase II engineering development tests at Paxtuxent River were aimed at continuing Phase I objectives and incrementally increasing each gain to establish the gain envelope. Final refinements to the system such as signal conditioning and the trip system adjustments were made. After the VSC configuration was frozen, the last task was the development of two VSC exercises.

Test Envelope

All tests were conducted within the NATOPS flight envelope for the SH-60B, as modified by the NAVAIR flight clearance. Hover tests were ground referenced and initially conducted at 300 feet AGL, which is above the single engine H-V diagram for the test aircraft. As confidence in the VSC system was gained, the tests were conducted at lower altitudes where the visual cueing was adequate for precision hover tasks. A minimum altitude limit of 20 feet AGL was observed for all tests.

Test Methods

General flying qualities test methods and procedures were in accordance with USNTPS "Rotary Wing Stability and Control" Flight Test Manual [4]. A specialized test method was the frequency sweep which was used to generate the frequency response data. This data was used to determine the gain and phase margin of the system which was the primary method for tracking the stability of the VSC system/aircraft combination.

Threat Analysis

Preflight analysis was performed to determine potential "threat" modes of failure and general hazards. The contractor conducted a System Safety Hazard Analysis [5] and several threats were identified. The threats or hazards are classified into two groups: Group 1 - threats or occurrences leading to a SAS actuator hardover; and Group 2 - threats or occurrences leading to an oscillatory behavior of the SAS actuator.

Failures Leading to Hardover

This failure mode could be the result of a VSC system failure or an aperiodic divergent aircraft mode. The effects of this failure mode were considered to be acceptable based on review of Sikorsky's SAS hardover flight tests, and USNTPS staff evaluations of SAS hardovers in a NASA UH-60 simulation. The VSC induced hardovers are no more critical than production SAS hardovers, and are equivalent to flying the aircraft with the SAS off, with a one inch bias in the longitudinal or lateral cockpit control positions. This failure mode is considered safe as long as the aircraft is operated with at least two inches of cyclic control margin.

Failures Leading to Oscillatory SAS Actuator

Several oscillatory modes were considered to be threats because of the possibility of relatively small

(± 10%) oscillatory SAS actuator motion exciting large amplitude rotor or structural motions. The degree of threat was considered to be related to the frequency of the mode as compared to the bandwidth of the VSC system. If the frequency of the mode being considered was significantly higher than the bandwidth of the system, the potential threat of the VSC to interact with the mode to cause a instability was low. If however the frequency of the mode under consideration was equal to or less than the bandwidth of the VSC system (including actuators), then the possibility of an adverse interaction between the VSC and the mode was treated as a real threat.

VSC System Bandwidth

In order for the VSC system to contribute to an oscillatory failure mode a signal from the VSC sensor would have to pass through the sensor, VSC computer, and SAS actuator. The bandwidth of this system is primarily limited by the bandwidth of the VSC computer and the SAS actuator.

The VSC computer bandwidth is determined by an analog lowpass second order filter with a bandwidth of 5 Hz. The filtering provided by the SAS actuator was determined by bench tests and verified during in-flight tests. These tests show that the SAS actuators can be modeled as fourth order with a bandwidth of approximately 5 Hz.

The combined VSC computer and SAS actuator form an effective sixth order low pass filter with a bandwidth (conservative) of 5 Hz. This frequency served as a rough discriminator of which excitation or modal frequency would be considered a threat.

Threat Modes

Surveys of SH-60 airframe structural data, general rotor dynamics literature, and interviews with subject matter experts highlighted several modes which represented potential threats. These modes are:

Lead-Lag Regressive - The rotor lead-lag regressive mode was identified [6] as a lightly damped mode that can couple with rigid body mode to produce an instability when an angular rate signal is fed back to the same axis rotor control (e.g. roll rate feedback to lateral cyclic). At the time of the test there were no data available to identify this effect for the SH-60B however an estimated value of the lead-lag regressive mode frequency was 3.2 Hz for the UH-60. This frequency was used as an initial estimate for the NSH-60B, and served to localize the search for instabilities during subsequent data analysis.

Tail Shake - A significant threat was identified as the tail shake mode. A survey of airframe structural data indicated a first lateral bending mode of the fuselage occurred at approximately 5 Hz and a first vertical bending mode at 6 Hz. Supporting this concern were NATOPS and fleet reports of a 5 Hz tail shake oscillation in certain flight conditions. This mode was treated as a significant threat relevant to feedback of yaw rate and attitude to the directional axis.

Others - Additional signal flow paths associated with VSC system crossfeeds (or coupling) were identified as lower level threats to stability. These crossfeed paths are enabled by aircraft design features such as the rotor hinge offset and the SH-60 canted tailrotor. There were no hard data available on these modes. However the knowledge of these coupling mechanisms focused attention during the analysis of the data from the crossfeed tests.

Tracking the Closed Loop Gain Margin

Various time and frequency domain stability tracking schemes were considered. The frequency domain method described below was proposed and approved by the NAVAIR flight clearance office.

The method uses baseline open loop frequency response data to estimate the limit VSC gain that can be used without exceeding the closed loop gain or phase limits of 6 db and 45 degrees respectively. The frequency sweeps were limited to a maximum magnitude of one inch and frequency of two Hz.

The major steps of this method are listed below. Example data from the roll axis are shown in figures 2 and 3.

- Lateral control frequency sweeps of up to one inch is generated by the pilot from a low frequency of approximately 0.5 Hz to a maximum of 2 Hz. (figure 2)

- The frequency response for the loop is determined. Both specialized and off the shelf spectral estimation algorithms were used.

- The open loop frequency response results are plotted in the standard Nichols form of gain on the y axis and phase on the x axis. Figure 3 shows the results for the example lateral sweep.

- The allowable gain for the loop is then estimated by measuring the vertical distance (in db) that the data can be slid vertically to meet the required stability margin.

- This gain, measured in db, is converted to a maximum or minimum value to be entered in the CCS. The value range for input to the CCS is 000 to 999 with 500 corresponding to zero gain.

The VSC gain is then changed towards the predicted gain limit, a closed loop frequency sweep performed and then the actual closed-loop frequency response is compared to the frequency response predicted from the open loop data.

If the agreement is good then the tracking method is considered validated, and the process is repeated until the limit gain is reached.

If the agreement is not adequate, the reason for the disparity is resolved. If the reason can't be resolved, the most conservative interpretation of the data is made and the process continued.

Off-Axis Effects

The above gain limit is considered valid for the aircraft configuration of the tests. This configuration includes the conventional parameters such as loading and weight, and also the other off-axis configuration parameters such as VSC gain and lead/lag values. The off-axis VSC gains along with helicopter cross coupling response may cause the predicted gain limit to change when off-axis gains are changed.

Since the off-axis effects are prevalent in the helicopter, the method of tracking the stability margin is subject to the above effects. These effects were in part responsible for the scatter band observed in the data. The VSC gain limits were estimated from the conservative (upper) side of the scatter band of the Nichols data thus reducing the magnitude of the predicted limit gain value. More refined approaches are available to address these off-axis effects [7]. These approaches will reduce the error band applied to the data and will "carve out" additional portions of the VSC gain envelope thus expanding the capability of the system.

Test Procedures

For all flights, variations of a specific gain were terminated if any one of the following conditions occur:

- Predicted closed loop gain margin of less than 6 db or 45 degrees
- The onset of any structural/rotor mode or excessive vibration as determined by onboard data analysis, air crew, or ground observer
- Unacceptable handling qualities (as indicated by a tendency towards an instability, excessive workloads or pilot induced oscillations)
- Sufficient gain level for instructional use
- Maximum or minimum gain setting (available on the CCS) was reached

A incremental progression to the maximum and minimum values was used to determine the limiting gain values (or combinations of gain settings) that could be used without encountering the above termination conditions.

Special Precautions

Special precautions used included:

- Hazard Analysis [5]
- Electro Magnetic Compatibility Safety Of Flight Tests (EMC SOFT)
- Strict adherence to termination conditions

- A NAVAIRWARCENACDIV Safety Checklist was completed.
- Gains were adjusted in an incremental fashion
- AUTOPILOT was OFF for all VSC operation (Except early AFCS fault monitoring tests)
- Real-Time VSC actuator monitoring

Test Results

The VSC gain ranges were determined during the Phase I and Phase II development tests and are presented in Table 1 and Figure 4. The gain levels in Figure 4, set at 500 (zero VSC gain), are presently assigned to less important VSC parameters, and were not developed. These excess channels, the result of using the Learjet VSC as the basis of for the SH-60B design, are channels available for future system growth.

The gain limits shown in figure 4 were based on several of the termination conditions listed previously. A summary of the gain limiting conditions is presented in Table 2. Gain limits in most cases determined by the minimum or maximum CCS settings. Other factors limiting the maximum gain were gain/phase margin, and for the directional axis with heading feedback, excessive saturation of the VSC actuator.

The overall VSC system's gain envelope is adequate for USNTPS flight demonstrations. The system represents a significant improvement over the capability of the NCH-46 VSC and the design architecture has excellent potential for growth.

VSC System Reliability

There was early concern that the VSC system designed for a fixed wing airplane would have poor reliability when installed in a helicopter. There were, however, no significant VSC system failures in over 100 flight hours of system operating time plus an additional 100 hr. of flight time with the system in a non-powered state. Two minor system failures occurred which required local maintenance action. They were a light bulb change and straightening of a bent VSC connector pin. Although the limited amount of data does not allow a statistically significant estimate of reliability, the trouble free operation to date suggests a highly reliable system.

CCS Operation

The management of the VSC system was evaluated by the instructor/safety pilots during ground and flight operation. The operation of the VSC system was via the CCS and associated control panels and cockpit switches. The instructor is able to quickly change VSC configurations by indexing through prestored VSC configuration gain sets. The time required to change configurations is approximately 5-10 seconds (the SOW allowed up to 30 sec.). The operation of the system required minimum attention away from the instructor's primary duties and will allow the maximum amount of training during each flight. The CCS operation represents a significant improvement over the NCH-46 VSC and is a enhancing feature of the NSH-60B VSC.

Trip System Performance

The VSC system can be disengaged (or tripped) from the SAS actuators by the safety trip system. This trip system has two modes: 1) the manual mode which allow either the instructor or evaluation pilot to trip the system, and 2) the auto safety trip mode which generates a trip independent of pilot action.

Manual Mode

The manual safety trip was checked prior to each flight and was used as the primary method to disengage the system. The manual safety trip never failed to work correctly and was never used to trip the system as the result of an adverse event such as an oscillatory divergence or VSC induced vibration.

Auto Mode

The auto safety trip system is activated by a difference between the commanded and actual SAS actuator position. If the difference is more than 4 volts for 100 ms, the safety trip system actuates and returns the control of the SAS actuator to the host aircraft. The above volts-time threshold values were used during all engineering development tests to date and represented a tradeoff between catching significant VSC failures, and producing unwanted trips or false alarms. During the development testing, the trip system thresholds were set at low values (a "hair trigger") which produced a high rate of false alarm safety trips. This high occurrence of false alarms was considered acceptable during development testing since an added degree of protection was afforded against unknown or miss-predicted instabilities. At the completion of the development phase the safety trip system thresholds were increased to allow representative mission tasks to be flown for the syllabus development flights.

There were no observed cases where the auto safety trip system made a significant save from an adverse occurence such as an oscillatory divergent actuator motion.

Description of VSC Flight Exercises

One interim and two syllabus VSC system exercises have been developed and are described below.

Interim VSC Exercise

A prototype VSC exercise was developed and presented to USNTPS staff and class. The gain envelope use was limited to that cleared to August 1993. The scope of the demonstration was a cross section of the VSC I and VSC II exercises described below.

VSC I

This exercise is the first helicopter VSC in the syllabus and is conducted in week 22 of the 48 week course. Academic instruction discussing the effects of sensitivity and damping on the response characteristics of the helicopter is followed by a simulator laboratory reviewing the effects discussed in the classroom and summarizing the basic test techniques used for inflight estimation of sensitivity and damping. A fixed task is flown while the VSC system is used to demonstrate a systematic variation of lateral control sensitivity and roll damping. A handling qualities evaluation is conducted where Cooper-Harper ratings are assigned and then analyzed in a post flight review.

VSC II

This exercise is conducted in week 29. Academic instruction discussing the sources of coupling is followed by a simulator laboratory reviewing the effects discussed in the classroom and specialized test techniques used for inflight estimation of control and rate coupling. A variety of fixed tasks are flown while the VSC system is used to demonstrate variations of control and rate coupling. Various response types and control delays are also demonstrated.

Future NSH-60B VSC Developments

VSC system enhancements, identified as being desirable to improve the effectiveness of the system, are:

- Increase the VSC system authority - will allow more aggressive mission tasks to be performed without saturation of the system
- Optimize the safety trip system - will allow a better balance between the protection afforded and the false alarm rate of the trip system
- Add other feedback variables - will allow demonstration of additional response type such as Translational Rate Command
- Add a variable feel system - will allow in-flight demonstration of the significant influence of the feel system on flying qualities
- Improve quality of existing VSC sensor signals - improvement of the quality of the input signals by sensor relocation and refinement of the signal conditioning will reduce the signal noise levels
- Improve the methods for testing and data analysis - will reduce the time and increase the accuracy of future VSC testing

Summary

Simulation has been used as a teaching tool at the USNTPS since the introduction of the VSC aircraft program in 1960. In-flight simulators have proven valuable for the teaching aircraft dynamics, controls and display system characteristics, and their combined effect on handling qualities.

Basic methods of implementing the

VSC were reviewed and past and current systems used at the USNTPS were discused. A general description of a response feedback VSC system was presented and the various subsystems were described. Criteria for selection of the host airframe, and the specific performance requirements of a VSC installed in a NSH-60B helicopter were presented. The VSC program elements of design, implementation, and testing were reviewed and potential improvements were summarized.

The overall effectiveness of the VSC system was validated by compliance with initial VSC system requirements, compliance with other engineering performance measures, qualitative evaluations of the test pilot/ instructors and the reactions and critiques of evaluation pilots.

The following VSC system features were considered enhancing:

- A trip system that always worked
- Trouble free (reliable) operation
- Low workload to manage the VSC
- Lack of engage transients
- Acceptable disengage transients
- Pilot comfort down to 20 feet AGL
- Adequate authority in the roll and pitch axis

The VSC provided a training facility that replaces and advances the capability of the NCH-46 VSC and provides growth potential for future enhancements.

Acknowledgments

This program was the result of the contributions of many team members, representing many organizations. A partial list of the team members is;

USNTPS Project Officers and Engineers:
Mike Mosher, MAJ Vick Thombs, LT Sid Hatcher, LT Kevin Switick, LT Glen Knaust
FTEG (Rotary Wing): Lew Fromhart
Calspan : Ralph Siracuse

Also instrumental were members of the USNTPS/DYNCORP maintenance team.

References

[1] Richards, Robert B., et al, Ground Based Simulation in Test and Evaluation Education, AIAA Flight Testing Conference, August 1992

[2] Mosher, M., Switick, K., Knaust, G., et al, RPT NO: TPS-RTR01-94, May 1994, Development of The Variable Stability System Installed In The NSH-60B Helicopter In Hovering Flight

[3] Calspan Proposal , VSC Helicopter VSC Replacement Program - Technical Proposal, 7 July 1991.

[4] U. S. Naval Test Pilot School Flight Test Manual No. 107, Rotary Wing Stability and Control, Preliminary, 30 June 1991.

[5] Calspan SH-60 TM no. 4, VSC Helicopter System Safety Hazard Analysis, April 1992.

[6] Tishler, M. B., "System Identification Requirements for High-Bandwidth Rotorcraft Flight Control System Design," Journal of Guidance, Control, and Dynamics, Vol. 13, No. 5 1990, pp. 835-841.

[7] Tishler, M. B., "Comprehensive Identification from Frequency Responses," Class Notes, January 1991

Table 1
VSC Gain Summary

GAIN NO.	Parameter	CCS MIN	CCS MAX		GAIN NO.	Parameter	CCS MIN	CCS MAX
1	Pitch Damping	0	999		33	Ped to Lat	0	999
2	Roll Damping	0	784		34	Col to Lat	0	999
3	Yaw Damping	0	999		35	q to Ped	200	780
4	Pitch Sensitivity	0	999		36	p to Ped	200	800
5	Roll Sensitivity	0	999		37	Long to Ped	0	999
6	Pedal Sensitivity	0	999		38	Lat to Ped	0	999
7	Pitch Attitude Feedback	450	677		39	Col to Ped	0	999
8	Roll Attitude Feedback	350	750		40	p to Long - Lead	500	500
9	Heading Feedback	450	700		41	p to Long - Lag	500	500
10	Long Lead	500	800		42	r to Long - Lead	500	500
11	Long Lag	500	800		43	r to Long - Lag	500	500
12	Lat Lead	500	800		44	Lat to Long - Lead	500	500
13	Lat Lag	500	800		45	Lat to Long - Lag	500	500
14	Ped Lead	500	800		46	Ped to Long - Lead	500	500
15	Ped Lag	500	800		47	Ped to Long - Lag	500	500
16	q Lead	500	500		48	q to Lat- Lead	500	500
17	q Lag	500	500		49	q to Lat - Lag	500	500
18	p Lead	500	500		50	r to Lat - Lead	500	500
19	p Lag	500	500		51	r to Lat - Lag	500	500
20	r Lead	500	500		52	Long to Lat - Lead	500	500
21	r Lag	500	500		53	Long to Lat - Lag	500	500
22	Long T Delay	20	999		54	Ped to Lat - Lead	500	500
23	Lat T Delay	20	999		55	Ped to Lat - Lag	500	500
24	Ped T Delay	20	999		56	p to Ped - Lead	500	500
25	p to Long	0	999		57	p to Ped - Lag	500	500
26	r to Long	250	800		58	Long to Ped - Lead	500	500
27	Lat to Long	0	999		59	Long to Ped - Lag	500	500
28	Ped to Long	0	999		60	Lat to Ped - Lead	500	500
29	Col to Long	0	999		61	Lat to Ped - Lag	500	500
30	q to Lat	250	650		62	Col to Ped - Lead	500	500
31	r to Lat	200	650		63	Col to Ped - Lag	500	500
32	Long to Lat	0	999		64	q to Ped - Lag	500	500

Table 2
Factors Limiting
Maximum VSC Gains

REASON FOR TERMINATION	PERCENT OF GAINS IN THIS CATEGORY
Reached Closed Loop Gain Margin	15
Structural/Rotor Mode, Vibration	0
Unacceptable Handling Qualities	0
Sufficient Gain for Instruction	24
Maximum or Minimum CCS Gain Setting	55

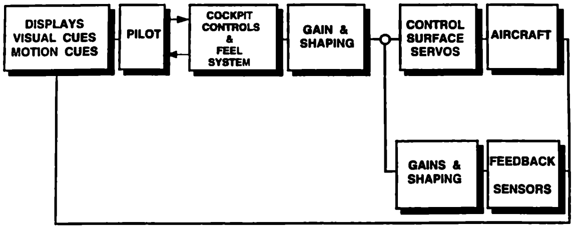


Fig. 1. Pilot-Vehicle System

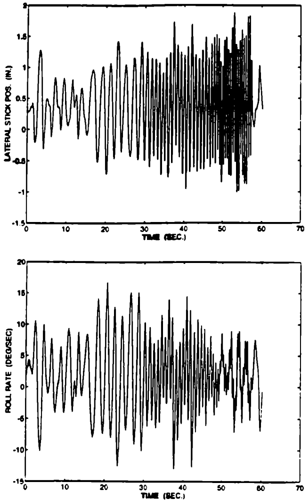


Fig. 2. Frequency Sweep

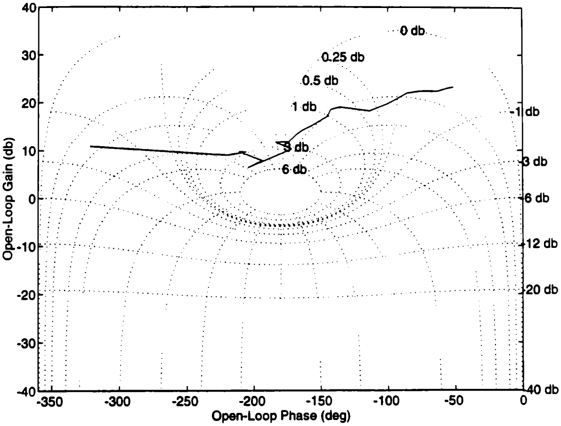


Fig. 3. Nichols Plot - Lateral Sweep

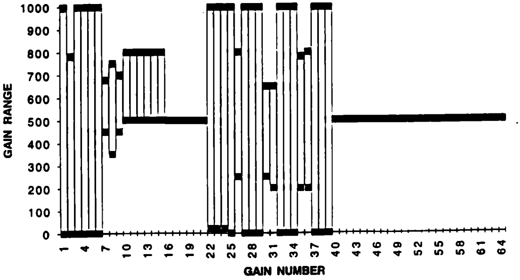


Fig. 4. Graphical Summary of VSC Gain Ranges

HELICOPTER SIMULATION FOR NAVAL TRAINING

D. LEDAR
Thomson Training & Simulation
Cergy-Pontoise, France

Abstract

This paper describes some key points in the tasks carried out by naval helicopter crew and the way in which the latest simulation technology can provide effective flight and mission training in areas not satisfactorily covered by most simulators to date.

The most significant training tasks are those which involve low altitude manual flight handling, such as take-off/ landing (including shipdeck landing), hover and transitions to flight, underslung load operation, SAR, ASW and ASSW missions.

The simulation of helicopter dynamics and the external environment has proved to be far more difficult than for fixed wing aircraft. However, the use of new cost-effective simulation technology allows the helicopter simulator manufacturer to overcome certain deficiencies.

1 - Simulator and naval helicopter crew training

Depending on the equipment fitted on the naval helicopter, the full mission training simulator system comprises a Full Flight Simulator (FFS) for the pilot, co-pilot and observer, and a Rear Crew Trainer (RCT) for the Tactical Coordinator (TACO) and the sonar and /or radar operator. The FFS is a dynamic device, mounted on a motion platform, and the RCT is a fixed base device. The two devices can be operated in either independent or integrated modes depending on the specific training tasks.

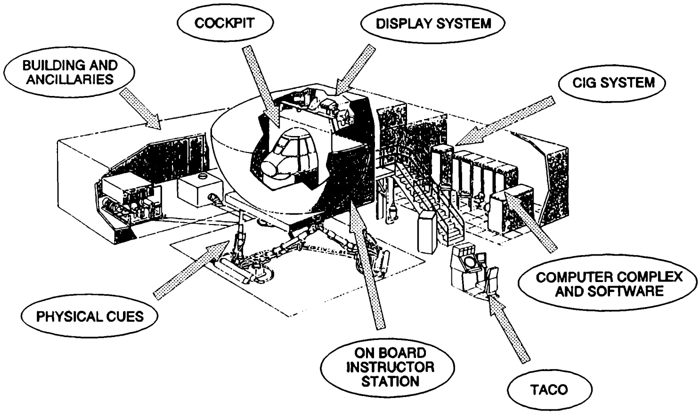


Fig. 1 - Simulator general view

Although the majority of the training tasks can be satisfactorily covered, there remains some deficiencies on most of the existing helicopter training devices, in the simulation of helicopter flight dynamics, the external environment and the visual scene. This situation leads to the requirement for some training to be still undertaken using the actual aircraft.

The most significant training tasks not satisfactorily covered by simulation to date are those which involve low altitude manual flight handling, such as take-off/landing, hover and transition to flight, underslung loads, winching, and, most significantly, shipdeck landing.

Flying costs are such that there is an increasing need for simulators which are not limited to flying training, but provide also tactical and mission training for the complete crew of the aircraft.

Customers wish more and more training tasks to be supported by a full mission simulator, which is a challenge to be taken up by the simulator manufacturer.

The extraordinary progress of simulation techniques, that has been performed in the last past years, allows now to satisfy most of helicopter operator requirements at a reasonable cost. Full SAR, ASW and ASSW mission training, including emergency procedures, can be covered by up-to-date simulators as those developed by Thomson Training & Simulation for Super Puma and Dauphin helicopters.

2 - <u>Naval helicopter crew training objectives</u>

Crew training for naval helicopters is usually oriented to the following tasks, some of which have been, up to now, more or less satisfactorily covered by simulation:

(a) <u>Helicopter Flight Handling</u>

- Take-off and landings : On land at an airfield or confined urban landing area. At sea on fixed (oil rigs) or moving platforms (ship-decks)
- Hover and transition to flight
- In flight manoeuvres
- Abnormal conditions : Ring vortex, blade stall, loss of tail rotor authority, malfunction conditions, emergency procedures
- Difficult weather conditions: High wind, turbulence, fog, cloud cover, icing, bad visibility, lightning, rain, storm, etc

(b) <u>Mission Training</u>

- Search & Rescue (SAR): Crew co-ordination, navigation, search patterns, homing, hover, winching, poor weather, high sea state
- Ship replenishment : Navigation, underslung loads, low speed and hovering on helideck flight handling
- Anti-Submarine Warfare (ASW): Crew co-ordination, low level hover (AFCS controlled) for sonar dip, MAD sensor operation, weapon deployment (torpedoes, depth charges)
- Anti Surface Ship Warfare (ASSW): Radar target search and lock-on, ESM/RWR surveillance, pop-up manoeuvres, missile launch and evasive manoeuvres, Over-The-Horizon-Targeting (OTHT), night-time missions with NVGs.

3 - Improvements in helicopter simulation techniques

The simulation of helicopter dynamics has proved to be far more difficult than for a fixed wing aircraft.

However, Thomson Training & Simulation is now in a position to propose new cost-effective solutions that overcome certain deficiencies:

(a) Flight handling

The flight handling qualities experienced by pilots on simulators often suffer from the latency of response to control inputs, causing Pilot-Induced-Oscillation (PIO) and also from the poor fidelity of the helicopter aerodynamics model, during certain phases.

The "critical flight loop" latency of response, measured for example between the flight control input and the change seen in the visual scene, has been reduced, thanks to the use of higher computational iteration rates, made possible by the use of more powerful RISC processors.

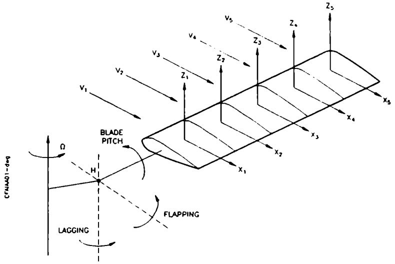
For accurate handling qualities, an effective motion system is essential to provide the correct stimuli to the pilot, particularly at low speed and hover, because small cues are perceived in the visual scene in that situation. In addition, the simulator has to include a vibrating platform to restitute with accuracy the actual aircraft behaviour.

The modelling of the aerodynamics has been improved by use of Blade Element Theory model (BET) for the main rotor and by comprehensive modelling of all the various contributions from the airframe, tail rotor, weapon pylons and other loads, wind, ground effect and interaction of the main rotor downwash on the other components.

The BET model, compared to classical models, provides the following features:

- each blade is modelled separately as a number of elements, taking into account the variable profile and twist blade
- local phenomena on the rotor disk are computed, such as ground effect on induced velocity, blade stall, flapping and lagging motion of the blades
- a theoretical induced velocity model computes the corresponding data in the whole flight envelope, including autorotation and vortex
- local malfunctions acting on the blades, such as icing, projectile impacts, dissymmetric blades, rotor out of track, are taken into account

The resulting quality depends on the number of elements per blade and the number of computations per rotor rotation, in order to have a correct representation of rotor forces and moments through the rotor disk. Satisfactory result is obtained with at least 5 elements per blade and 30 computations per rotor rotation.



Where :

H	is the hinge
Z_i	is the lift of element i
X_1	is the drag of element i
V_1	is the local airflow vector
Ω	is the rotor speed

Fig. 2 - Breakdown of a blade

(b) External environment

The simulation of the external environment concerns the world outside the helicopter and the modelling of sensory effects. It includes radio stations, the atmospheric conditions, sound and motion effects. It has been up to now inadequate to cover some helicopter training tasks.

The modelling of wind, gusts and turbulences when flying close to structures has not been realistic. Simulation of wind, gusts and turbulence levels close to structures, such as ship and oil rig towers, or sea-cliffs, is made now more realistic by application of empirical models based on pilot experience. The purpose of the modelling is to vary the workload of the pilot and provide realistic real-world conditions during certain training tasks. The response of the helicopter to the environmental wind effects is integrated within the aerodynamic model.

Finally, shipdeck landing training was excluded from the simulator capabilities since the simulation of ship motion has not been modelled satisfactorily. The improvement of ship motion simulation is obtained by computation of movement in pitch, roll, yaw and heave using a complex sinusoidal movement based on measurements from actual ship types in various sea states and wind conditions. The sea can be modelled in 3 dimensions (3D) to present ship movement correlated with sea and to take into account real ship characteristics.

(c) Visual generation and display

The displayed visual scene often lack both the realism and Field Of View (FOV) to allow effective low altitude flight close to the landing zone or ship.

The development of ever higher performance visual image generators now allows high levels of detail and realism to be programmed into the visual data base by using phototexture techniques. The pilot can now be given scene realism which allows accurate estimation of aircraft height, distance and attitude cues that were previously inadequate. In particular, enhanced sea texture details and downwash effect provide altitude and speed cues, allowing low altitude flying above sea. In addition, bow, stern and wake waves provide the necessary cues to assess the right speed of the landing ship. The 3D modelling of sea state and moving ship are controlled by software to provide realistic, real-world ship motion. Multiple point feedback of the height and slopes of the surface under the helicopter allows accurate undercarriage contact with the shipdeck. At the final approach phase, display of the rotor blades within pilot's visibility plot has been deemed as a necessary cue when evolving in a confined area.

Mission training requires, for a realistic restitution of the visual environment, a high detail modelled terrain representation, which must be completed by a great number of 3D modelled objects like: boats, submarines, missiles, other helicopters for formation flying etc. These objects are "living", which means, for example, that landing gear or rotor of an helicopter are animated.

The new real time Computed Image Generator (CIG) developed by Thomson-CSF allows its simulators to benefit from state-of-the-art image generation technology at reasonable cost. In particular, such CIG offers extensive photographic texturing with micro- textures. Accurate representation of real site is obtained by use of actual terrain photographs.

Texture rendering in tactical areas and landing zones requires both a geo-specific photographic representation and a very high resolution (less than 20 cm). This is obtained by extending the resolution of specific phototextures by a microtexture modulation computed in real time. The pilot's eye gets naturally and steadily acquainted with dominant texture detail in the scene, ranging from the low altitude approach where specific photographic elements are used as visual markers, up to the touchdown where microtexture details allow the pilot to keep evaluating altitude, speed and attitude.

At last, the use of high update rates also ensures smooth visual scene update even during high rate yaw manoeuvres.

Visual display systems have evolved to allow continuous and large covering of the vertical and horizontal field of view required for certain helicopter manoeuvring, such as shipdeck landing. Direct projection with the latest high technology video projectors, using seamless edge matching and distortion correction, onto a large partial dome provides the continuous scene. Using a direct projection, rather than a collimated system provides the pilot with a more realistic distance perspective during low altitude and landing manoeuvres. For most training, the visual point is set-up to the central cockpit position, but for phases like shipdeck landing, the eye point can be switched to the pilot seat, in order to give the student the accurate view of the deck and the glidepath indicator.

Particular attention has to be paid to the vertical field of view which is of primary importance during landing, take-off and hovering phases.

To achieve the goal of a motion compatible visual system with large field of view and high resolution image, Thomson-CSF has developed a low weight, high stiffness spherical screen able to cover FOV as large as 200° horizontal and 100° vertical. Thomson-CSF has also developed a new version of CRT projector, called Phebus 5, which is a raster calligraphic, 9" CRT projector with HDTV capability.

The high brightness and unique resolution capacity of Phebus 5 enable a large FOV to be covered with a smal number of projectors.

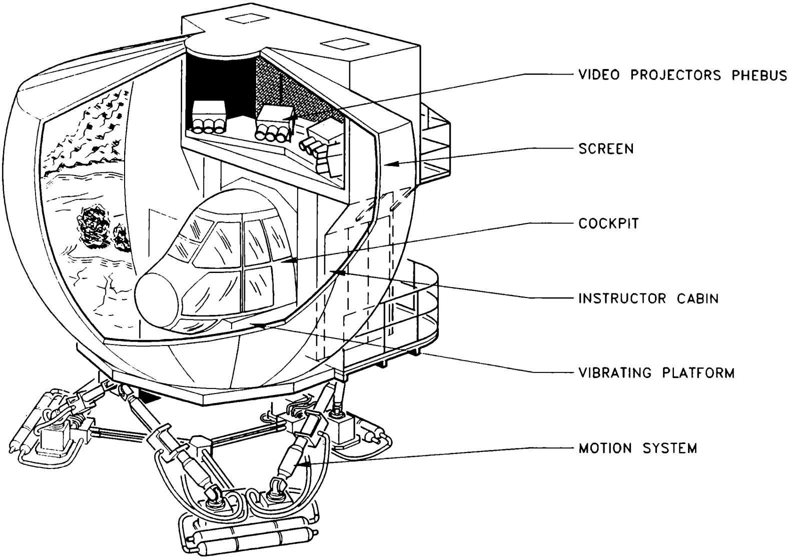


Fig. 3 - Visual display system

(d) Mission scenarios

The impressive increase of computer and workstation power and mass storage capacity, combined with the availability of standard and dedicated sofware tools, like data base and graphic management software products, allows now to provide simulators with a comprehensive tactical environment for complex mission scenario training.

A scenario can include a wide range of friend or enemy moving objects of any type: aircraft, ground vehicle, ship, submarine, etc, with pre-programmed course and equipped with missile, gun and active counter-measure system. The scenario, previously prepared, is selected within a library and progresses automatically during the training session, allowing the instructor to devote all his attention to crew monitoring. A state of the art instructor station, with graphic interactive man-machine interface is installed in the flight compartment to support instructor's monitoring.

4 - Shipdeck landing training

Shipdeck landing is an area where the fidelity of the simulation has been lacking. This is mainly due to limited visual representation, inadequate FOV and latencies of response. Visual details and cues, including: ship motion and waves, deck details, sea state are very important, to bring enough information to the pilot for height, speed and attitude evaluation when landing on the shipdeck.

Effective training in shipdeck landings can now be addressed by high fidelity simulation, thanks to accurate flight handling, state-of-the-art visual generation and display systems and realistic environment simulation as: wind, turbulence around structures and 3D sea correlated to ship motion.

Comprehensive emergency procedure training, like engine fail during take-off or landing, at day or night, ditching with use of floatation gear, can be accomplished with incomparable safety conditions and lower cost than using actual helicopter.

The training task can be broken down into a number of stages, each of which place specific demands on the simulation fidelity.

MISSION PHASE	VISUAL SIMULATION CUES	OTHER SIMULATION FEATURES
Recovery	• Ship recognition from all approach directions • Variable sea state • Wind velocity • Other vessels	• Navigation to ship (radar,tacan,nav. computer)
Circuit	• Ship's wake • Drift assessment	• Large horizontal visual FOV, cross cockpit view
Approach	• Glide-path indicator + Ghost Amber light • Horizon bar • Mast head lights • Stern lights ± 22,5 ° • Accurate ship proportions • Hangar & hangar roof details	• Effects of turbulence in ship wake • Accurate flight dynamics and motion
Landing	• Ship heave, Pitch, Roll, Yaw • Deck markings • Detail on port side • Flight deck officer and wands • Deck flood lights • Helicopter shadow on deck	• Accurate HAT visual feedback • Motion & sound cues for touchdown and harpooning • Ship motion simulated after touchdown • Effects of turbulence & IGE/OGE
Take-off	• Same as for landing	• Same as for landing

5 - Complex scenarios for mission training

(a) Overview

The purpose of a scenario is to locate the helicopter in a tactical environment consisting of enemy, allied and neutral objects. They can be fixed objects, moving on preprogrammed courses or manually controled.

The helicopter crew, airborne in his environment, is able to use the operational equipment at his disposal to observe, interprete the tactical situation and act in compliance with its objectives. Management of the scenario is performed by the computer, in real time, concurrently with the execution of the helicopter simulation programs. There is consequently a high degree of coherence between the radar and visual images, on one hand, and between the motion of the targets and the helicopter, on the other hand. Likewise any disturbance introduced by the instructor, a malfunction in the helicopter for example, or turbulence, rough sea state, poor visibility, etc, in the environment, results in a degradation of the weapon system operating conditions, and thus an increase in exercise difficulty.

In the full mission simulators newly developed by Thomson-CSF, the scenario progresses automatically, which alows instructor to focus on crew monitoring, but he can override any of the preprogrammed actions and take manual control of a target. The overall result, as perceived by the ownship, is a mission training environment which incorporates threats and friendly forces employing tactics which correspond to the training objectives of the scenario, chosen from a library of a hundred ones.

(b) Composition of a scenario

A scenario is defined by a real or imaginary geographical gaming area consisting in coast line with recognizable landmarks. A dozen moving targets with pre-programmed courses, and up to 50 fixed targets are located in the gaming area. Targets can be enemy, allied or neutral and are high fidelity models of the following types: merchant ships, frigate, cruiser, helicopter carrier, submarines, fixed and rotary wing aircraft, dinghy, box, ground vehicles. A target can be assigned: trajectory, speed, acceleration, diving speed, radar reflectivity, magnetic susceptibility and be equipped with a weapon system (radar, missiles, gun), electronic counter measures (ECM) equipment, etc.

Each of the above elements is customized according to user's needs. It has, where applicable, a realistic response to the changing environment, i.e. under different situations, depending on the best strategy. This highly modular and functionnal system is served by a powerful scenario creation tool, which runs on an independant workstation and allows the user to create a wide library to cover a whole range of missions and situations, including: SAR, ship replenishment, ASW and ASSW.

(c) Scenario monitoring

The instructor monitors the execution of the scenario using the on-board instructor station which displays graphic screens:

- tactical situation map showing the coast lines, position of ownship and objects
- ownship navigation parameters
- object parameters such as: speed, altitude, distance from ownship
- status of the weapon system
- results of firing
- repetition of crew's radar screen

A hard copy system provides the capability to record tactical situations encountered during the training session, for use by the instructor to support the debriefing with the crew.

During the execution of a scenario, the instructor is able to modify the scenario, namely:

- activate or deactivate any threat
- control manually a selected moving object trajectory by using facilities of displays such as: situation map, cockpit view presenting a synthetic fish-eye view from the object controlled, associated with main instruments (ball, rate of climb, speed, altitude)
- force AAA or missile firing of a given threat
- activate an agressive trajectory

At any time of the exercise the instructor can freeze the progress of a scenario, and comment the progress of the exercise.

(d) Examples of scenarios

Such scenario management system is the key tool that complete the helicopter simulator to permit full training for SAR, ASW and ASSW missions. Examples of summarized scenarios are given hereafter.

SAR scenario

"After take-off upon alert, in surface situation mode, search and rescue operation, navigation, patterns, weather advoidance, are followed by AFDC coupler mode, hover and hoisting of survivor, and concluded by casualty evacuation to medical facility".

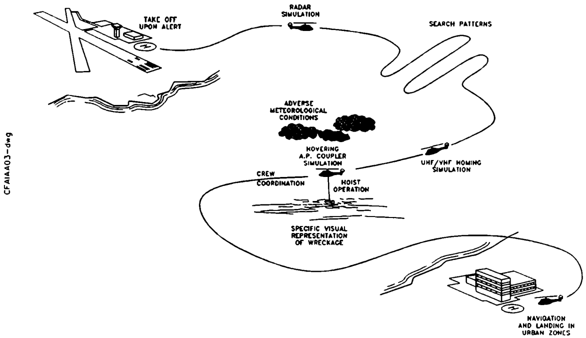

TAKE OFF UPON ALERT
RADAR SIMULATION
SEARCH PATTERNS
ADVERSE METEOROLOGICAL CONDITIONS
HOVERING A.P. COUPLER SIMULATION
UHF/VHF HOMING SIMULATION
CREW COORDINATION
HOIST OPERATION
SPECIFIC VISUAL REPRESENTATION OF WRECKAGE
NAVIGATION AND LANDING IN URBAN ZONES
CFAIAA03-dwg

Fig. 4 - SAR mission

ASW scenario

"Take-off from ship, flight with tactical situation and coordination with friendly units, integrated search patterns, radar + MAD operation are ended-up by launch of torpedoe and return to ship".

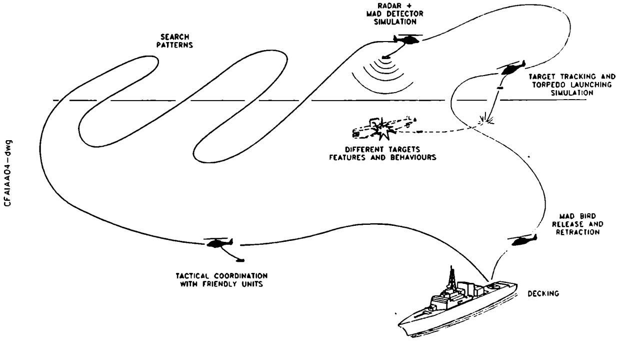

SEARCH PATTERNS
RADAR + MAD DETECTOR SIMULATION
TARGET TRACKING AND TORPEDO LAUNCHING SIMULATION
DIFFERENT TARGETS FEATURES AND BEHAVIOURS
MAD BIRD RELEASE AND RETRACTION
TACTICAL COORDINATION WITH FRIENDLY UNITS
DECKING
CFAIAA04-dwg

Fig. 5 - ASW mission

ASSW scenario

"Take-off from land base, tactical flight with radar and missile operation are concluded by firing the missile to target, air attack on other elements, using gun, and return to home base".

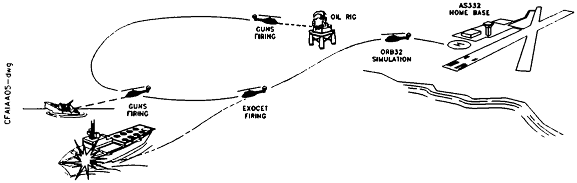


Fig. 6 - ASSW mission

6 - Conclusion

Simulator has demonstrated to be the up-to-date equipment to support flight training, because it is safer, cheaper and has better availability rate than using actual aircraft.

The technology available today, combined with the experience developed in simulation principles, allows now the manufacturing of helicopter simulators which meet most of naval training needs.

Continuous progresses in: computers, CIG, aircraft and environment modelling should permit to approach soon the "zero flight time training" already performed with fixed wing aircraft simulators.

Thomson-CSF is currently under contract with a Naval Force to build full mission simulators for two helicopter types: Super Puma and Dauphin, both of which offer full mission training capabilities.

AIAA-94-3436-CP

A BLADE ELEMENT MODEL IN A LOW COST HELICOPTER SIMULATOR

Kenneth Graham, Jr.*, and Amnon Katz †.
Department of Aerospace Engineering
The University of Alabama
P. O. Box 870280, Tuscaloosa, AL 35487-0280

ABSTRACT

The equations for a blade element helicopter rotor model are presented. The model was developed at the University of Alabama Flight Dynamics Lab for use in real-time and off-line simulation. This blade element model simulates the trajectories of the individual rotor blades and allows the representation of some transient effects that are not included in some other models.

I INTRODUCTION

Bladehelo is a model of a helicopter rotor, that simulates the motion of each individual blade. Drawing on recent advances in microprocessors, this can now be done with modest means. Blade- helo is being developed at The University of Alabama Flight Dynamics Lab (UA FDL). Off line, Bladehelo is an analysis tool. It has been applied in research into the performance benefits of higher harmonic control[1]. Running in real time, Bladehelo, makes for a low cost, high fidelity man-in-the-loop simulator.

The disk models of earlier practice[2] estimated the forces and moments imparted by the blades to the hub averaged over one revolution. Many simplifying assumptions were needed to carry this estimate out analytically. Some of those were removed with the advent of computers. Still, disk models operate under the following two premises. The first is that the averaging process is carried out for a steady state condition of the rotor that corresponds to the instantaneous or delayed control inputs. The second is that the response of blade motion to control inputs is assumed to be linear. These assumptions are dispensed within Bladehelo. We still average over a revolution in the off line work. We average over a quarter revolution in real time, but the rotor blades go through transients and naturally seek and find the steady state when appropriate. No linearity is imposed. The full non-linear mechanical equations apply. This allows higher harmonic responses to swashplate control inputs to the extent that they occur.

Disk models are adequate for many purposes, but the technology of digital computation is at a stage where it no longer makes sense to explore and define the limits of applicability of disk models. The blade model is cheap enough for general use.

II OVERVIEW

The current version of Bladehelo assumes a rigid blade. Only flapping motion is allowed. The flapping hinge is located at the hub. It is felt that this arrangement is adequate for performance studies. Analysis of handling qualities, however, requires an offset flapping hinge. In the real time version, an offset is "simulated" by introducing an appropriate moment into the hub (see below).

The forces and moments on the blades fall into the categories of inertial/gravitational and aerodynamic. The inertial/gravitational portion for each blade can be calculated in closed form using the moment of inertia and center of gravity of the blade. The moment of inertia and center of gravity are determined from blade input data at initialization.

The blade is described in an input file as a sequence of elements. The mass, span, chord, and geometric twist of each element are specified, and the lookup table to be used for aerodynamic data is indicated. This allows blades with varying cross sections to be analyzed.

The aerodynamic force is computed in real time, element by element, using the element aerodynamic data, twist angle, and local velocity. The local angle of attack is determined, and a table appropriate for the local sèction and Mach number is searched for lift and drag coefficients. Total force and moment are accumulated by summing over the elements.

The history of the flapping angle, β, is integrated numerically. Its second derivative, $\ddot{\beta}$, is determined from the condition that the moment around the flapping hinge vanish. The history of β, in turn, influences both aerodynamic and in-

* Graduate student.

† Professor, member AIAA.

ertial forces. Any net forces and moments not balanced within the blade are transferred to the hub.

LIST OF SYMBOLS

$\vec{a}$	acceleration of a point on the blade relative to the inertial frame
A	lateral cyclic input
$\vec{a}_b$	acceleration of the body relative to inertial space
$\vec{a}_o$	acceleration of the origin of blade system relative to the inertial frame
B	longitudinal cyclic input
C_l	section lift coefficient at a cross section of the blade
C_d	section drag coefficient at a cross section of the blade
$\vec{d}$	drag per unit span of a cross section of the blade
$\vec{e}_s$	unit vector along the blade
$\dot{\vec{e}}_s$	rate of change of $\vec{e}_s$ relative to the blade system
$\ddot{\vec{e}}_s$	rate of change of $\dot{\vec{e}}_s$ relative to the blade system
$\vec{F}$	total force applied to the hub by a blade
$\vec{f}_{aero}$	aerodynamic force per unit span at a point on the blade
$\vec{F}_{aero}$	total aerodynamic force applied to the hub by a blade
$\vec{f}_{iner}$	inertial/gravitational force per unit span at a point on the blade
$\vec{g}$	acceleration of gravity
$\vec{h}$	position of the rotor hub relative to the body system
$\vec{i}, \vec{j}, \vec{k}$	unit vectors along the $x, y,$ and z blade axes
$\vec{I}, \vec{J}, \vec{K}$	unit vectors along the $X, Y,$ and Z body axes
I_f	moment of inertia of a blade about its flapping hinge
$\vec{l}$	lift per unit span of a cross section of the blade
m	mass of a blade
$\vec{M}$	total moment applied to the hub by a blade
$Mach$	Mach number of a cross section of the blade
$\vec{M}_{aero}$	total aerodynamic moment applied to the hub by a blade
N	number of elements along a blade
Q	mass flow rate of air through the rotor
$\vec{r}$	position of a point on the blade relative to the blade system
R	length of a rotor blade
$\dot{\vec{r}}$	velocity of point on the blade relative to the blade system
$\ddot{\vec{r}}$	acceleration of point on the blade relative to the blade system
r_{cg}	distance from hub to the center of gravity of the blade
$\vec{v}$	velocity of a point on the blade relative to the free stream air mass
$\vec{V}$	component of the velocity of a point on the blade relative to the local air which is in the plane of the cross section
$\vec{v}_a$	velocity of a point on the blade relative to the local air
$\vec{v}_b$	velocity of the helicopter center of gravity relative to the air mass
$\vec{v}_i$	downwash velocity induced by the rotor
$\vec{v}_o$	velocity of the origin of the blade system relative to the free stream air mass
V_s	speed of sound in air
x, y, z	blade system axes
X, Y, Z	body axes
α	angle of attack of a cross section of the blade
β	blade flapping angle
$\dot{\beta}$	flapping angular rate
$\ddot{\beta}$	flapping angular acceleration
θ	total pitch angle of a cross section of the blade
θ_c	pitch control input to the blade
θ_o	collective pitch input to the blade
θ_t	geometric twist of a cross section of the blade
ρ	density of air
Ψ	blade azimuth angle
$\vec{\omega}$	angular velocity of the blade system relative to the inertial frame
Ω	rotational rate of the rotor shaft relative to the body
$\dot{\Omega}$	rate of change of Ω
$\dot{\vec{\omega}}$	angular acceleration of the blade system relative to the inertial frame
$\vec{\omega}_b$	angular velocity of the body relative to the inertial frame
$\dot{\vec{\omega}}_b$	angular acceleration of the body relative to the inertial frame
$\vec{\omega}_r$	angular velocity of the blade system relative to the body

III ROTOR FORCES AND MOMENTS

We introduce a blade coordinate system with its origin at the rotor hub. The z axis points down along the rotor shaft; the x axis extends along the projection of the rotor blade onto a plane normal to the shaft; and the y axis completes the right hand triad. $\vec{i}$, $\vec{j}$, and $\vec{k}$ are unit vectors along the x, y, and z axes of the blade system.

We also use a helicopter body system of coordinates with its origin at the center of gravity of the body and its Z axis parallel to the shaft, pointing down. The body X axis points forward in the fuselage plane of symmetry, and the Y axis points to the right. The unit vectors along the body axes are denoted $\vec{I}$, $\vec{J}$, $\vec{K}$. The body and blade axis systems are aligned when the rotor azimuth angle, Ψ, is zero. At other times, the transformation from the body system to the blade system consists of a single left hand rotation about the z axis through the azimuth angle Ψ. Referencing Ψ to the X axis, the upwind direction, is contrary to the common practice of referencing it to the downwind direction. This was done for the convenience of having the axis systems aligned when the azimuth angle is zero. These axis systems are shown in Figure 1.

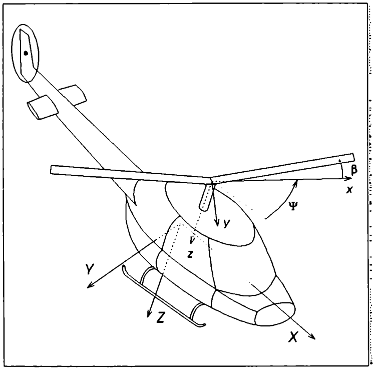


Figure 1: Coordinate Systems and Conventions

To calculate the forces on an element of the rotor blade, it is necessary to determine its velocity and acceleration. Taking the translation and rotation of the blade system into account, the velocity relative to the free stream air mass is given by

$$\vec{v} = \vec{v}_o + \dot{\vec{r}} + \vec{\omega} \times \vec{r}. \tag{1}$$

The air mass is assumed to be an inertial reference frame. Therefore, the acceleration relative to the inertial reference frame is

$$\vec{a} = \vec{a}_o + \ddot{\vec{r}} + 2\vec{\omega} \times \dot{\vec{r}} + \dot{\vec{\omega}} \times \vec{r} + \vec{\omega} \times (\vec{\omega} \times \vec{r}). \tag{2}$$

In these equations, $\vec{r}$ is the position of a point on the blade relative to the blade system and $\vec{\omega}$ is the total angular velocity of the blade system. $\vec{v}_o$ and $\vec{a}_o$ are the velocity and acceleration, respectively, of the origin of the blade coordinate system. These are also relative to the inertial free stream air mass reference frame and are given by

$$\vec{v}_o = \vec{v}_b + \vec{\omega}_b \times \vec{h}, \tag{3}$$

$$\vec{a}_o = \vec{a}_b + \vec{\omega}_b \times (\vec{\omega}_b \times \vec{h}) + \dot{\vec{\omega}}_b \times \vec{h}, \tag{4}$$

where $\vec{v}_b$, $\vec{a}_b$, $\vec{\omega}_b$, and $\dot{\vec{\omega}}_b$ are the velocity, acceleration, angular velocity, and angular acceleration of the helicopter body axis system relative to the inertial frame. $\vec{h}$ is the position of the origin of the blade coordinate system, relative to the origin of the body coordinate system.

The total angular velocity, $\vec{\omega}$, and total angular acceleration, $\dot{\vec{\omega}}$, of the blade system must also be determined. The angular velocity of the blade system relative to the body system is

$$\vec{\omega}_r = -\Omega \vec{k}. \tag{5}$$

Ω is equal to $\dot{\Psi}$, the time rate of change of the azimuth angle. This is also the rotational rate of the rotor shaft relative to the body. The angular velocity of the blade system relative to inertial space is

$$\vec{\omega} = \vec{\omega}_b + \vec{\omega}_r. \tag{6}$$

Then, the angular acceleration is given by

$$\dot{\vec{\omega}} = \dot{\vec{\omega}}_b - \dot{\Omega}\vec{k} + \vec{\omega}_b \times \vec{\omega}_r. \tag{7}$$

The remaining vectors in equations (1) and (2), $\vec{r}$, $\dot{\vec{r}}$, and $\ddot{\vec{r}}$, describe the motion of a point on the blade relative to the blade coordinate system. This motion is a rigid rotation around the flapping hinge with an angular velocity $\dot{\beta}\vec{j}$. If we define $\vec{e}_s$ to be a unit vector along the blade, the position of the point of interest on the blade is

$$\vec{r} = r\vec{e}_s. \tag{8}$$

The unit vector is given by

$$\vec{e}_s = cos\beta\,\vec{i} - sin\beta\,\vec{k}. \tag{9}$$

Because the blade is rigid, r, the distance from the hub to the point of interest, is constant. Therefore, the velocity of the point relative to the blade system is

$$\dot{\vec{r}} = r\dot{\vec{e}}_s, \tag{10}$$

and the acceleration relative to the blade system is

$$\ddot{\vec{r}} = r\ddot{\vec{e}}_s. \tag{11}$$

$\dot{\vec{e}}_s$ is given by

$$\dot{\vec{e}}_s = \dot{\beta}\vec{j} \times \vec{e}_s, \tag{12}$$

and, likewise, $\ddot{\vec{e}}_s$ is

$$\ddot{\vec{e}}_s = \ddot{\beta}\vec{j} \times \vec{e}_s + \dot{\beta}\vec{j} \times \dot{\vec{e}}_s. \tag{13}$$

As noted above, $\ddot{\beta}$, which is contained in equation (13), is required to integrate the history of the blade flapping angle, β, numerically. $\ddot{\beta}$ is obtained by observing that the sum of the moments about the flapping hinge must be zero. The sum of the moments about the hinge is represented by

$$\int_0^R (\vec{r} \times \vec{f}_{iner}) \cdot \vec{j}\,dr + \vec{M}_{aero} \cdot \vec{j} = 0. \tag{14}$$

$\vec{f}_{iner}$ is the inertial/gravitational force per unit span and $\vec{M}_{aero}$ is the moment on the blade caused by aerodynamic forces.

For the purpose of evaluating the aerodynamic forces and moments, the blade is divided into N finite elements. For the current rotor model, the aerodynamic properties are considered to be constant along the span of each element. If the position vector, $\vec{r}_n$, and the aerodynamic force per unit span, $\vec{f}_{aero\,n}$, are determined at the center of the nth element, the aerodynamic force on a blade, and the flapping moment caused by that force, are given by

$$\vec{F}_{aero} = \sum_{n=1}^{N} \vec{f}_{aero\,n} \Delta r_n \tag{15}$$

$$\vec{M}_{aero} = \sum_{n=1}^{N} (\vec{r}_n \times \vec{f}_{aero\,n}) \Delta r_n. \tag{16}$$

To evaluate these equations, $\vec{f}_{aero}$ must be calculated. The aerodynamic force per unit span is found by determining the velocity through the air and the angle of attack of the cross section through the point of interest. The angle of attack along with the local Mach number are then used, with the lookup table for the particular element, to determine the lift and drag coefficients. These are then used to determine the magnitude of the lift and drag per unit span.

First, the velocity of a point on the blade relative to the local air, $\vec{v}_a$, is required. $\vec{v}$, given by equation (1), is the velocity relative to the free stream air. However, there is an additional component of velocity induced by the force exerted on the air by the rotor. The downwash induced by the rotor is computed by momentum theory as

$$\vec{v}_i = -\frac{\vec{F}_{aero}}{Q}, \tag{17}$$

where, here, $\vec{F}_{aero}$ is the force exerted on the entire rotor by the air mass and Q is the mass flow rate of the air which is given by

$$Q = \pi R^2 \rho |\vec{v}_o + \tfrac{1}{2}\vec{v}_i|. \tag{18}$$

In this equation, R is the radius of the rotor blades and ρ is the density of air. Only half of $\vec{v}_i$ is effective at the rotor, therefore, the velocity of a point on the blade relative to the air at the rotor is

$$\vec{v}_a = \vec{v} - \tfrac{1}{2}\vec{v}_i. \tag{19}$$

To determine the angle of attack, α, for an element, it is necessary to determine the portion of $\vec{v}_a$ that is in the plane of the blade's cross section, perpendicular to the span. The component of $\vec{v}_a$ along the span is assumed not to contribute to the aerodynamic forces. The component of the velocity in the plane of the airfoil is given by

$$\vec{V} = \vec{v}_a - (\vec{v}_a \cdot \vec{e}_s)\vec{e}_s. \tag{20}$$

Figure 2 depicts the cross section of the blade and the in-plane velocity.

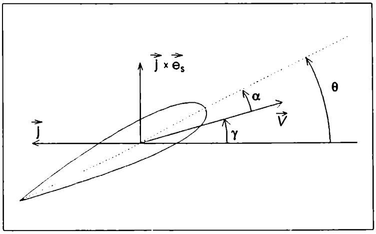


Figure 2: Blade Angle of Attack

Next, we wish to determine the angle between the in-plane velocity and the negative y axis. This angle is given by

$$\gamma = atan2(\vec{V} \cdot (\vec{j} \times \vec{e}_s),\ -\vec{V} \cdot \vec{j}). \qquad (21)$$

In this equation, the *atan*2 function of computer practice is employed to assure a correct value of γ over the full range of angles from $-\pi$ to π. The angle of attack of the section is given by

$$\alpha = \theta - \gamma, \qquad (22)$$

where θ is the pitch angle of the cross section, defined so that, at zero pitch, the chord is parallel to the y axis with the leading edge facing in the negative y direction.

The pitch of the airfoil section is, in turn, given by

$$\theta = \theta_t + \theta_c. \qquad (23)$$

θ_t is the geometric twist of the cross section of the blade and is specified in the input data for each element. θ_c is the pitch control input to the blade. For a typical swashplate arrangement, the pitch control input is

$$\theta_c = \theta_0 + A\, cos\Psi + B\, sin\Psi, \qquad (24)$$

where θ_0 is the collective pitch input, A is the lateral cyclic input - positive to the right, and B is the longitudinal cyclic input - positive forward.

Now that the angle of attack has been determined, the local Mach number is required. This is given by

$$Mach = \frac{V}{V_s}. \qquad (25)$$

where V_s is the speed of sound and V is the magnitude of the vector $\vec{V}$.

The angle of attack and the Mach number can now be used to determine the lift and drag coefficients, C_l and C_d, from the lookup table associated with the element. By definition, the lift of an airfoil is perpendicular to the velocity and the drag is parallel to the velocity. Therefore, the lift and drag per unit span of the element are

$$\vec{l} = \tfrac{1}{2}\rho c C_l V(\vec{e}_s \times \vec{V}), \qquad (26)$$

$$\vec{d} = -\tfrac{1}{2}\rho c C_d V \vec{V}, \qquad (27)$$

where c is the element cord length.

Finally, the aerodynamic force per unit span on an element is

$$\vec{f}_{aero} = \vec{l} + \vec{d}. \qquad (28)$$

This can now be used for each element in equations (15) and (16) to determine the aerodynamic force and moment on a rotor blade.

We next wish to calculate the inertial/gravitational force and moment on the blade. Integration of the inertial/gravitational portion of (14) is carried out over the length of the blade. As previously discussed, the assumption of a rigid blade allows this to be done in closed form. The mass of the blade, m; the distance from the flapping hinge to the center of gravity of the blade, r_{cg}; and the moment of inertia of the blade about the flapping hinge, I_f, are required, and are determined from the input data for the blade elements.

The inertial/gravitational force on a differential element of the blade is given by

$$\vec{f}_{iner}\, dr = dm(\vec{g} - \vec{a}), \qquad (29)$$

where $\vec{g}$ is the acceleration of gravity and dm is the mass of the differential element. $\vec{a}$ is given by equation (2), which incorporates (11), which, in turn, involves the value of $\ddot{\beta}$. By substituting equation (29) into (14) and expanding, the integration of the inertial portion can be carried out analytically and an equation for $\ddot{\beta}$ can be obtained. With the components of vectors in the blade coordinate system denoted by the subscripts 1, 2, and 3, the equation for $\ddot{\beta}$ is

$$\begin{aligned}\ddot{\beta} = \frac{m\, r_{cg}}{I_f}&\left[(a_{o\,1} - g_1)\, sin\,\beta + (a_{o\,3} - g_3)\, cos\,\beta\right] \\ &+ \tfrac{1}{2} sin(2\beta)\left(\omega_1^2 - \omega_3^2\right) \\ &+ \omega_1\omega_3 cos(2\beta) - \dot{\omega}_2 + \frac{M_{aero\,2}}{I_f}.\end{aligned} \qquad (30)$$

$\ddot{\beta}$ is used to integrate the time history of the flapping angle β.

With β and its first and second time derivatives known, the total force and moment transmitted to the hub by the blade is given by

$$\vec{F} = \int_0^R \vec{f}_{iner}\, dr + \vec{F}_{aero}, \qquad (31)$$

$$\vec{M} = \int_0^R \vec{r} \times \vec{f}_{iner}\, dr + \vec{M}_{aero}. \qquad (32)$$

These equations are expanded and the integration along the span is again carried out analytically. The resulting expressions are

$$\begin{aligned}\vec{F} =& m[\vec{g} - \vec{a}_o - r_{cg}(\ddot{\vec{e}}_s + 2\vec{\omega} \times \dot{\vec{e}}_s + \dot{\vec{\omega}} \times \vec{e}_s \\ &+ \vec{\omega} \times (\vec{\omega} \times \vec{e}_s))] + \vec{F}_{aero},\end{aligned} \qquad (33)$$

$$\begin{aligned}\vec{M} =& \vec{e}_s \times [m r_{cg}(\vec{g} - \vec{a}_o) - I_f(\ddot{\vec{e}}_s + 2\vec{\omega} \times \dot{\vec{e}}_s \\ &+ \dot{\vec{\omega}} \times \vec{e}_s + \vec{\omega} \times (\vec{\omega} \times \vec{e}_s))] + \vec{M}_{aero}.\end{aligned} \tag{34}$$

IV COMPUTATIONAL PROCEDURE

Section III derives the equations governing the flapping motion of the rotor blades – one degree of freedom per blade. These equations are to be applied in conjunction with the six degree of freedom equations for the helicopter fuselage[3]. It is these equations that govern the quantities $\vec{v}_b$, $\vec{a}_b$, $\vec{\omega}_b$, and $\dot{\vec{\omega}}_b$ – the velocity, acceleration, angular velocity, and angular acceleration of the helicopter body axis system relative to the airmass. The coordinate transformation from earth coordinates to helicopter body coordinates also comes from the body equations. This transformation is a necessary ingredient in determining the components of $\vec{g}$ in equation (30).

The numerical evaluation of equation (30) calls for substitution of the angular velocity of the blade system, $\vec{\omega}$, from equation (6), and the angular acceleration, $\dot{\vec{\omega}}$, from equation (7). $\vec{a}_o$, the acceleration of the origin of the blade coordinate system is determined by equation (4). The aerodynamic moment, $\vec{M}_{aero}$, is given by equation (16), with $\vec{f}_{aero}$ coming from equation (28). Equation (28) in turn depends on equations (17) through (27).

The blade flapping acceleration $\ddot{\beta}$ of equation (30) is integrated to find the flapping velocity $\dot{\beta}$, which, in turn, is integrated to find the new flapping angle β. The previously determined values are substituted into equations (33) and (34), along with $\dot{\vec{e}}_s$ given by equation (12) and $\ddot{\vec{e}}_s$ given by (13), to find the total force and moment transmitted to the hub by a blade.

The entire procedure just described is carried out for each of the rotor blades. Then, the force and moment transmitted to the hub by all blades are included in the integration of the body equations of motion to obtain the new state of the helicopter.

V IMPLEMENTATION

The equations described in section III allow the trajectory of the rotor blades relative to the helicopter to be determined for a particular state of the helicopter body. In turn, they also yield the forces and moments exerted by the blades on the rotor shaft, which are then used to integrate the state of the helicopter body.

Due to the high rotational rate normally associated with the helicopter rotor, relatively small time steps are required for the numerical integration of the trajectory of the rotor blades. A much longer time step is adequate for the integration of the equations of motion of the helicopter body. For this reason Bladehelo uses *small steps* to integrate the blade trajectory and *large steps* to integrate the state of the body. Each large step consists of an integral number of small steps. The forces and moments on the rotor blades used in the body equations are averaged over all small steps within the particular large step.

The current model assumes that the rotational rate of the rotor, Ω, is constant. Therefore, the large and small time steps also represent fixed increments in terms of the azimuth angle. We found it best to make the large steps divisible into a revolution. This eliminates aliasing effects due to loads that cancel over a revolution.

Bladehelo is currently being implemented in two forms. Off-line, the state of the helicopter body is assumed to remain constant. This tool is currently being used for evaluation of the effects of higher harmonic control of the rotor on the power required by the rotor and on the top speed that is sustainable by the helicopter[1]. Since the equations of motion of the body are not integrated for this implementation, the size of the large time step has no effect on the motion of the blades. The large step is set equal to an entire revolution. The forces and moments, averaged over a revolution, are used to determine performance.

The second implementation of Bladehelo is in the UAFDL real time helicopter simulator. It is being developed for use in man-in-the-loop simulation to study performance and handling qualities of rotors and rotor blade control systems. For this application, a large step corresponding to a quarter of a revolution has been found to provide a reasonable trade-off between accuracy and computational efficiency.

This setup allows a high fidelity real-time simulation to be performed with modest means. Currently, execution of the rotor model and the helicopter dynamics is done by a Supercard with an i860 microprocessor, hosted in a PC. Each simulation frame, the i860 processor executes the flight model and communicates the state of the helicopter to the PC. The PC then handles communications with the computer systems that handle image generation and instrument displays. This communication is done in parallel with the execution of the flight model for the next frame.

During initial development of Bladehelo, a simplified model of the helicopter body (Figure 3) was employed. This helped isolate the effects of rotor details. The simplified body is modeled as a sphere on the rotor shaft line. The only aerodynamic force on the body is drag, proportional to the square of the velocity, acting through the center of the sphere. The inertial/gravitational properties of the sphere are expressed in terms of mass and moment of inertia (which, however, retains the non spherical ellipsoid of inertia of a realistic helicopter body). Also, the tail rotor is not modeled. Instead, the pedals control the body yaw rate directly. As the rotor model is further refined, models of a full helicopter body are being evaluated.

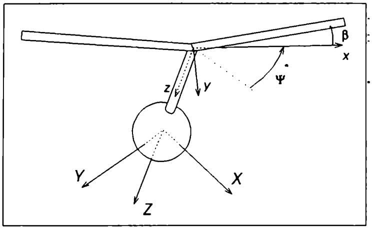


Figure 3: Simplified Helicopter body

The current equations for Bladehelo were developed for flapping hinges attached directly to the center of the hub. The lack of offset should have little effect on the performance of the rotor. A rudimentary autopilot rigged for the off-line studies had no trouble controlling this rotor. Yet, for manned simulation, the zero offset rotor proved difficult to control. Offset hinges transmit a moment to the shaft that tends to align the helicopter body with the rotor. This makes control easier (although it does induce increased stresses in the shaft).

The effect of offset hinges was "simulated" in the current model, by introducing a moment along the blade system y axis, although no changes were made in the calculation of the blade trajectory and forces. The magnitude of the applied moment is equal to the force applied to the hub by the blades in the z direction times the offset distance. Future versions of Bladehelo will have offset hinges.

V CONCLUSIONS

A high fidelity blade element model has been presented that is suitable for real-time simulation with modest resources. This model accounts for transients, blade stall, and higher harmonic effects. The model is well suited for analysis of rotor design and for evaluation of advanced rotor control methods such as individual blade and higher harmonic rotor control systems. The model can be used off-line as well as in real time simulation.

The preliminary model has already been employed for rotor performance studies[1] and some real-time simulation. Many refinements are planned to improve the accuracy and computational efficiency. These include offset flapping hinges, optional teeter hinge and lead/lag hinges, and tapered blade elements. Finally, improved methods for determining the inflow velocity distribution will be investigated.

REFERENCES

1. Katz, A., "Performance Benefit of Second Harmonic Control in Helicopters," to be published in the Journal of Aircraft.
2. Bramwell, A. R. S., *Helicopter Dynamics*, John Wiley & Sons, New York, 1976.
3. Katz, A., "Notes on Dynamics" Course notes, The University of Alabama, Tuscaloosa, 1994.

AIAA-94-3438-CP

TOWARDS MORE REALISM IN NVG USE IN TRAINING SIMULATORS

P. RAPP
Thomson Training & Simulation
Cergy-Pontoise, France

Abstract

The paper presents original solutions made up by Thomson Training & Simulation , a subsidiary of Thomson-CSF, that largely improves the realism of simulated NVG images. In particular, improved contrast and simulation of specific effects related to bright light point sources are discussed.

1. Introduction

Recent developments in visionics and sensor techniques, in particular FLIR and image intensifiers, now allow the users of both civilian and army helicopters to operate their aircraft in more and more degraded visibility conditions. Indeed, all present and future transport and combat helicopters are or will be fitted with facilities for tactical flight, navigation and engagement in all-weather and/or night-time conditions.

Yet, in spite of the rapid and spectacular enhancement of those new facilities, a growing number of night-flight accidents has highlighted the limitations of such systems and consequently shown that:

- the crews are not fully aware of the operating and/or training conditions,
  safe crew training means should be found.

The answer to this new requirement is given by simulation, whose most recent developments allow the realistic simulation of the sensors involved. Thomson Training & Simulation, refered to as TTS hereafter, through a comprehensive control of the whole visual chain, is in a position to provide original answers to any simulation problem.

The purpose of this paper, which deals with the particular case of the Night Vision Goggles (NVGs), is to:

1) explain the NVG simulation problem,
2) present the various alternatives and the choices made for the simulators already delivered or those being developed by TTS.

2. Notes on the NVGs

2.1. Operation

For each eye, the NVGs are composed of an image intensifier that amplifies the residual light of a night scene, as coming from the moon, the stars and artificial lighting, and an optical device meant to display the thus-generated image to the user.

The image intensifier is a photomultiplier which transforms and amplifies the photon intensity into electrons that hit a phosphorescent layer, generally green. State-of-the-art NVGs (this paper only discusses the 3rd-generation NVGs), in addition to a better suitability to the night images spectrum (the 3rd-generation NVGs cover the spectrum ranging about 500 to 900 nm as shown in Fig. 1), feature the automatic control of the photomultiplier's luminance gain relatively to the incoming light source in order to get a more or less constant output level, hence prevent phosphor saturation as much as possible.

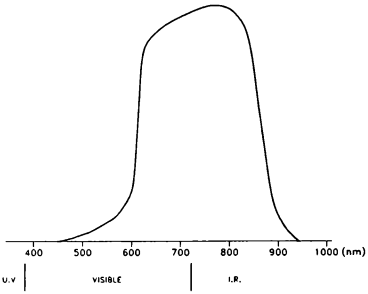


Fig. 1 Typical spectral response of Gen III NVGs

2.2. Characteristics of the night scenes

The residual light amplified by the NVGs comes from the moon and the stars as well as from artificial lighting (towns, roads, airports, etc.). The natural light is a near infrared (IR) light; the lower part of the spectrum is concerned by the moonlight, whereas its upper part conveys the starlight. The intended light levels approximately range from less than 0.5 mLux to over 100 mLux (very dark to very clear night). Moreover, the night-time reflectivity of materials is different from the day-time one since it mainly depends on the wavelength. Fig. 2 shows a few examples of materials. You can see for instance that the concrete reflectivity/grass reflectivity ratio is inverted from the lower part to the upper part of the spectrum.

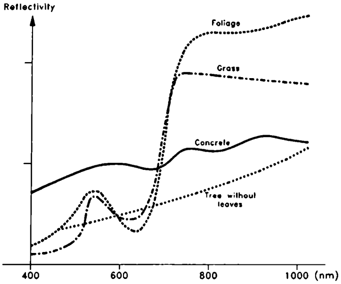


Fig. 2 Material reflectivity

2.3. Operational use

As opposed to the image intensifiers, the FLIR sensor does not require any residual light, but it is much more expensive, heavy and uneasy to implement. Besides, in thermal images, contrast is noticeably degraded after heavy rain- or snowfalls.

On the contrary, image intensifiers are quite easy to implement and, owing to their small volume and lightness, can be fitted directly on pilot's helmets. Yet, some of their limited performances restrict their operational use or require precautions. Actually, when using NVGs:

- minimum residual light is required,
- the field of view is narrow (about 40°),
- range adjustment is not always available, which impairs close-range vision,
  the resolution is low,
  flying over low-contrast landscapes is risky,
  the image easily gets saturated by bright light sources.

Moreover, the cockpit requires some adjustments, particularly as regards instrument lighting, which should be made compatible with NVG sensitivity.

Finally, certain armies use filtered NVGs, which reject the lower spectrum. The solutions adopted for cockpit instruments lighting compatibility differ accordingly. These rejection filters operate more or less high in the spectrum, as shown in Fig. 3. Whereas the French Army uses the whole spectrum, the German and British Armies use filters at respectively 645 and 650 nm, and the spectrum in American NVGs is cut off at either 625 or 665 nm depending on the operating conditions.

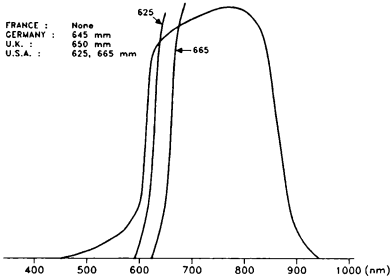


Fig. 3 NVGs with filters

3. Importance of training

The poor visibility/low-level flight combination obviously helps the helicopter crew to surprise the enemy or protect themselves from threats, but also puts them in difficult, even hazardous operating conditions. That is why it is extremely important that the crews should be trained for controlling the aircraft in such extreme situations.

Present training, ensured mainly during real night flights, is intended to provide the trainees with the necessary reflexes and knowledge while teaching them to avoid traps and possibly fatal errors.

Reflexes include the permanent scanning of the landscape to compensate for the narrow NVG field of view. Doing so allows the trainee to integrate any element in his peripheral field of view and detect any moving object. Thus, the pilot's tendency to stare at an object can be corrected and the tunnel effect, which prevents correct speed estimation, can be compensated for.

Training also aims at making pilots aware of the night-time myopia that affects most of them, as well as of the limited resolution performance of the NVGs. One regrettable consequence of this latter point is that electric wires are detected too late during low-level flights. This danger is increased by the fact that distances are often overestimated, so that objects are seen as being farther or smaller than they really are.

Training for using the NVGs over such low-contrast areas as deserts, snow or water is also of paramount importance and extremely hazardous in real flight conditions.

Also important is the pilot's ability to correctly interpret night-time images, whose textures and contrasts can be quite different from those seen in daytime conditions. Identifying the nature of the landing terrain is essential. Let us also mention the perception of the information from the instruments with NVG-compatible lighting.

In addition to tricky obstacle clearance manoeuvres, pilots will have to learn how to take account, even advantage, of situations from violent lights to dark areas that may considerably impair their perception of the environment.

Finally, the coordination between the crew members who wear NVGs and those who do not requires specific training procedures.

However, training during real night flights, let alone aircraft cost and availability, presents some limitations. Obviously, for reasons of safety, it is not possible to get as close to the extreme conditions as what may be experienced during real missions. Actually, training flights are authorized only if certain minimum conditions are combined, including illumination, weather, flight altitude, visibility range, etc. Training for NVG flight also requires extremely qualified instructors.

An alternative to night training is daytime crew training with neutral densities added in front of the NVGs. Unfortunately the image seen by the trainee is not quite realistic especially as regards peripheral vision, light points, instruments perception or even contrasts.

In fact, only the full mission simulator is able to provide a safe and effective solution to the above-mentioned limitations and drawbacks.

4. NVGs and full mission simulator

4.1. Particular problems

NVG simulation presents a number of problems mainly due to the specific operating conditions and/or the design of the image generation and display systems, namely:

- NVGs are not necessarily worn by all crew members,
- the NVGs used differ from one country/army to another,
- the spectrum of the light from the visual systems used in simulation does not match that of natural night light, especially as regards infrared,
- the databases and the luminance processing should be revised to take account of the particular reflectivity of materials and the nature of the light sources at night,
- the blooming/halo phenomena cannot easily be reproduced without real NVGs.

4.2. TTS solutions

4.2.1. Baseline configuration, use of real or simulated NVGs

The full mission simulators make use, and will do so for another few years, of direct projection- or projection/collimation-type image display systems. In those conditions, one may choose between two options for NVG images simulation:

- the first one consists in projecting onto the screen an image whose colour, contrast and luminance characteristics are as close as possible to those of an image as seen through the NVGs. The limited field specific to the NVGs is then rendered by the use of model NVGs. But the same image can be seen with and without NVGs and besides, the peripheral image is not realistic,
- the second one consists in projecting an image with characteristics as close as possible to a night-time image and using real NVGs to watch it.

The 2nd solution has been retained by TTS for simulating the NVGs on simulators already or being delivered as it has major advantages over the 1st one, in particular:

- the light level of the image seen by non NVG wearers is more true to life,

this solution allows the specific NVG properties to be used for simulating certain phenomena that are uneasy, even impossible, to simulate otherwise such as halo and saturation,
real cockpit lighting is used normally.

4.2.2. Displaying a night image

The point here is to project an image with spectrum, contrast, light points, etc. characteristics as close as possible to those encountered by night for the pilot wearing NVGs as well as non NVG wearers to see realistic images.

PHEBUS, a projector specially developed by TTS for simulation purposes, is used for creating images that remarkably fulfill these requirements. PHEBUS is a TV/calligraphic projector with extreme flexibility allowing separate adjustments of the light levels of surfaces and light points. Moreover the red tube gives out sufficient luminous flux within the wavelength range compatible with that of the NVGs, even those with filters. Fig. 4 shows the emission spectra of the green, blue and red tubes of PHEBUS. To simplify, only the main lines and the red upper secondary line are shown. Practically, it may be considered that either the left part of the spectrum (moonlit night) or the right part of the spectrum (starry night) will be available. Experiments have proved that there is enough intensity in the main or secondary line as emitted by the red tube of PHEBUS to provide the necessary light intensity.

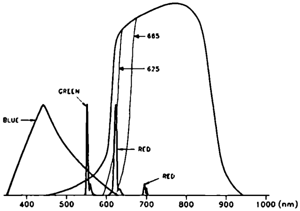


Fig. 4 PHEBUS spectral emission

4.2.2.1. Simulating contrast/reflectivities/cast shadows

Contrast is essential, e.g. for realistic tactical flight training. Contrast reproduction, which obviously depends on the dynamics of the luminance signal controlling PHEBUS, also depends on the differences in reflectivity between the various materials included in the scene. So it is important that the colour palette should be adapted to reality (see examples in Fig. 2). At a pinch one could imagine that surfaces with no red component in their visible spectrum become visible at night in the near IR region. That is why colour distribution over surfaces must be revised according to the reflectivity curve.

So, achieving maximum realism would require as a minimum one palette corresponding to a moonlit night, and one palette corresponding to a starry, moonless night.

Moreover, tactical flights over hilly or mountainous regions require the taking into account of the cast shadows, as those create very low, even null, reflectivity and contrast areas the positions of which depend on that of the moon.

4.2.2.2. Simulating the light levels of surfaces

Since a sufficiently high contrast on surfaces is required, there is no question of calculating the light levels of surfaces in the image generation system, by making use of the "natural" night-time ambient and directional luminosities. Saving the contrast level is done by bringing the image up to a daytime level, which allows PHEBUS to be operated in the best contrast conditions. Now, the light levels of surfaces may be dimmed in two different ways, i.e. optically or electronically.

Optical dimming is better than electronic dimming because it does not alter the contrast level of the generated surfaces. Indeed, with optical dimming, the black level is dimmed to the same proportions, whereas it remains identical with electronic dimming. So, optical dimming should be used for enhancing contrast quality.

However, let us point out that optical dimming also affects the light points, the levels of which should remain unchanged. So, light point simulation shall not be performed using the red tube (see paragraph 4.3).

Of course optical dimming, which can be obtained through densities or diaphragms, will depend on the luminance level required for the surfaces. In particular, the higher the rejection by the NVG filter, the lesser the dimming, at the risk of making surfaces visible to the naked eye.

4.2.2.3. Simulating light points

The simulation of light points is particularly important. As a matter of fact, the light points participate in the overall illumination of the scene, in the visible spectrum as well as in the near IR region to which NVGs are sensitive.

When fairly bright light points enter the NVG picture, the luminous flux received by the NVGs increases and the gain is consequently tuned down, which may impair the contrast of the image observed and create halo and blooming phenomena which increase image corruption.

We should also consider the occasional dazzle caused by weapon effects, such as explosions, missile firing, etc. This latter phenomenon will be easily simulated through a flash-type external light source.

The problem here is to display light points so that they can be seen by non NVG wearers with all their colour components and a light level high enough for the NVGs to be operated in realistic conditions. Only calligraphic light points allow this. PHEBUS, owing to its calligraphic capability, is a perfectly suitable solution.

4.3. Simulating the halo phenomenon with PHEBUS

The halo phenomenon takes place when the gain times light level product exceeds the maximum luminance the phosphor can take in without generating saturation (see Fig. 5), as is the case with particularly bright light points. Since the gain is inversely proportional to the luminous flux, we get low contrast on surfaces combined with brightening areas around the light points. Reproducing this in simulation requires realistic luminance conditions, that is a big difference in luminance between the surfaces and the light points. This dynamics can be created only through calligraphic light points.

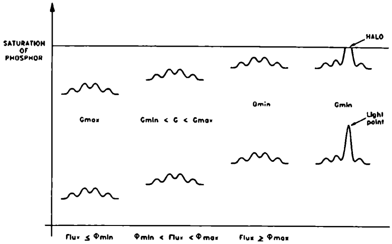
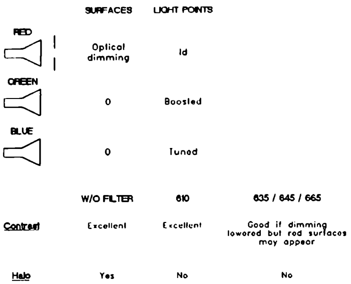


Fig. 5 Principle of halo phenomena

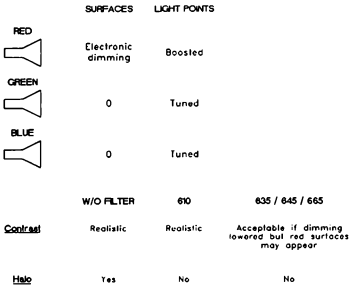
TTS has successfully reproduced the halo phenomenon in visual systems featuring projection or projection/collimation systems. The experiments have been carried out using gen. III NVGs (SOPELEM CN2H) plus a set of filters (610, 635, 645 and 665 nm) as similar as possible to those used in foreign army NVGs. Of course the best result (contrast, halo, realism) is obtained when making the most of the main line of PHEBUS's red tube (about 620 nm), as is especially the case when using NVGs without filters. Actually, the IR flux available from the main line is sufficient for dimming the red tube optically. The light points from the green tube are then used for simulating the halo.

Non NVG wearers do not see anything, except of course the light points. The higher the cut-off frequency of the filter, the lower the flux in the main red line. It is consequently necessary to rely on the secondary line, which is at about 690 nm. Optical dimming can no longer be used (since it also dims the light points) in the red tube since the halo phenomenon is then possible only when using the light points from the red tube. The weakening luminous flux will require lowered dimming of the red component in the projected image, although a red image may be seen by non NVG wearers. As examples, Fig. 6 and Fig. 7 show the results obtained during the experiments conducted on a projection/collimation-type display system. The retained solution will quite naturally depend on the NVGs involved and on the importance given to either contrast quality or halo simulation.



	SURFACES	LIGHT POINTS	
RED	Optical dimming	Id	
GREEN	0	Boosted	
BLUE	0	Tuned	
	W/O FILTER	610	635 / 645 / 665
Contrast	Excellent	Excellent	Good if dimming lowered but red surfaces may appear
Halo	Yes	No	No

Fig. 6 Solution favouring the contrast



	SURFACES	LIGHT POINTS	
RED	Electronic dimming	Boosted	
GREEN	0	Tuned	
BLUE	0	Tuned	
	W/O FILTER	610	635 / 645 / 665
Contrast	Realistic	Realistic	Acceptable if dimming lowered but red surfaces may appear
Halo	Yes	No	No

Fig. 7 Solution favouring the halo

4.4. Precautions

NVG simulation requires the fulfilment of certain conditions, without which simulation performance and realism might be impaired.

First, the cockpit shall be particularly well protected against spurious lights. This also applies to the on-board instructor's station, for which a simple curtain will perfectly do.

It is also necessary to make sure that the optical transmission capability of the simulated canopy is compatible with the performance of the projectors used.

Finally, edge mateling shall be as good as possible, since a bad overlay may be quite disturbing when seen through NVGs. Then again PHEBUS, owing to it exceptional performance, is the ideal solution to the problem.

5. TTS simulators with an NVG capability

Out of the four recently-ordered, new-generation helicopter simulators (one of which has already been put into service), three have an NVG crew training capability. These simulators are intended for three different Clients with varied requirements and operating contexts. Since the components of the visual system (CGI and display) also differ from one project to another, this gives TTS some unique experience in the new domain of NVG image simulation.

6. Conclusion

TTS's strategy has always been and is still the exclusive development, in particular for aircraft simulators, of full-feature and consistent visual systems from the computer image generator and databases to the whole range of image display systems: projection, projection/collimation and helmet-mounted visual systems. The technical development for NVG image simulation quite naturally comes within the framework of this policy. This led TTS to develop original solutions to the problems inherent with NVG image simulation.

Moreover, the considerable short-term requirements have convinced TTS to carry on the research and development effort to meet and anticipate at best the most stringent specifications.

The recent order for full flight/full mission simulators including the NVG capability is a definite proof of the faith the Clients have in the competence of TTS.

Author Index

Author Index

Author Index

Author Index